

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-232646

(43)Date of publication of application : 22.08.2000

(51)Int.Cl.

H04N 7/24
H04N 7/08
H04N 7/081

(21)Application number : 11-031945

(71)Applicant : SONY CORP

(22)Date of filing : 09.02.1999

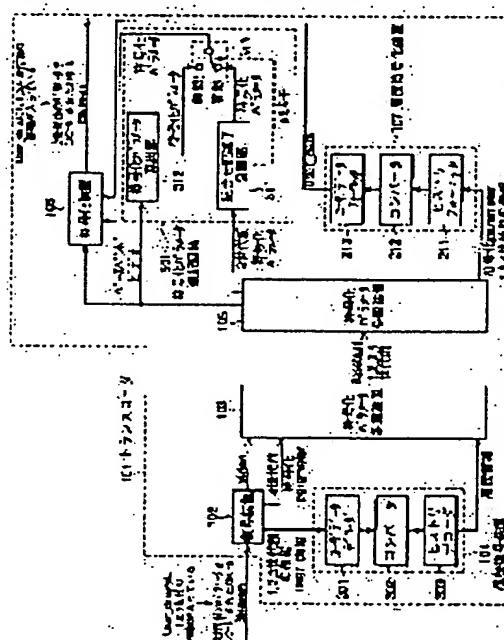
(72)Inventor : KITAMURA TAKUYA

(54) METHOD AND DEVICE FOR TRANSMITTING STREAM AND SERVED MEDIUM

(57)Abstract:

PROBLEM TO BE SOLVED: To transmit a stream with less picture deterioration due to re-encoding through a medium of a small capacity while reducing the scale of a transcoder.

SOLUTION: A coding parameter separator 105 extracts a coding parameter optimum to re-encoding and outputs the parameter to a combination descriptor separating section 511 together with a descriptor. A switch 513 selects either the coding parameter supplied from the combination descriptor separating section 511 or the coding parameter calculated from base band video data by a coding parameter calculation section 512 corresponding to the descriptor supplied from the combination descriptor separate section 511, and outputs the selected coding parameter to an encoder 106. The encoder 106 encodes the base band video data on the basis of the received coding parameter, multiplexes user-data including history information supplied from a history coder 107 and outputs the multiplexed data.



LEGAL STATUS

[Date of request for examination] 14.05.2002

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's]

BEST AVAILABLE COPY

decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号
特開2000-232646
(P2000-232646A)

(43)公開日 平成12年8月22日(2000.8.22)

(51)Int.Cl. ⁷	識別記号	F I	テーマコード(参考)
H 0 4 N	7/24	H 0 4 N	Z 5 C 0 5 9
	7/08		Z 5 C 0 6 3
	7/081		

審査請求 未請求 請求項の数5 O L (全 60 頁)

(21)出願番号 特願平11-31945

(22)出願日 平成11年2月9日(1999.2.9)

(71)出願人 000002185

ソニー株式会社

東京都品川区北品川6丁目7番35号

(72)発明者 北村 卓也

東京都品川区北品川6丁目7番35号 ソニ
ー株式会社内

(74)代理人 100082131

弁理士 稲本 義雄

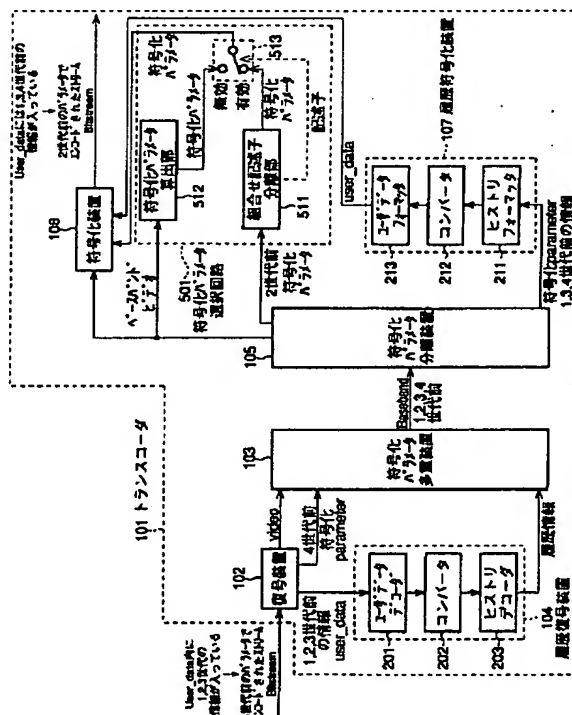
最終頁に続く

(54)【発明の名称】 ストリーム伝送装置および方法、並びに提供媒体

(57)【要約】

【課題】 トランスコードの規模を小さくするとともに、再符号化に伴う画像の劣化を抑制可能なストリームを、少ない容量のメディアを介して伝送できるようにする。

【解決手段】 符号化パラメータ分離装置105は、再符号化に最適な符号化パラメータを抽出し、記述子とともに、組合せ記述子分離部511に出力する。スイッチ513は、組合せ記述子分離部511から供給された記述子に対応して、組合せ記述子分離部511から供給された符号化パラメータ、または、符号化パラメータ算出部512がベースバンドビデオデータから算出した符号化パラメータを選択し、符号化装置106に出力する。符号化装置106は、入力された符号化パラメータに基づいて、ベースバンドビデオデータを符号化し、履歴符号化装置107から供給される履歴情報を含むuser_dataを多重化し、出力する。



【特許請求の範囲】

【請求項 1】 所定のメディアを介して符号化ストリームを伝送するストリーム伝送装置において、

過去の符号化処理において使用された複数の符号化パラメータから所定の符号化パラメータを選択的に組み合わせる組み合わせ手段と、

前記組み合わせ手段により組み合わせられた前記符号化パラメータの履歴を表す符号化履歴情報を生成する生成手段と、

前記生成手段により生成された前記符号化履歴情報を、前記符号化ストリームに重乗して前記メディアに伝送する伝送手段とを含むことを特徴とするストリーム伝送装置。

【請求項 2】 前記符号化ストリームは、MPEG方式で符号化されていることを特徴とする請求項 1 に記載のストリーム伝送装置。

【請求項 3】 前記重乗手段は、前記符号化履歴情報を、前記符号化ストリームにuser_dataとして重乗することを特徴とする請求項 1 に記載のストリーム生成装置。

【請求項 4】 所定のメディアを介して符号化ストリームを伝送するストリーム伝送装置のストリーム伝送方法において、

過去の符号化処理において使用された複数の符号化パラメータから所定の符号化パラメータを選択的に組み合わせる組み合わせステップと、

前記組み合わせステップの処理により組み合わせられた前記符号化パラメータの履歴を表す符号化履歴情報を生成する生成ステップと、

前記生成ステップの処理により生成された前記符号化履歴情報を、前記符号化ストリームに重乗して前記メディアに伝送する伝送ステップとを含むことを特徴とするストリーム伝送方法。

【請求項 5】 所定のメディアを介して符号化ストリームを伝送するストリーム伝送装置に、

過去の符号化処理において使用された複数の符号化パラメータから所定の符号化パラメータを選択的に組み合わせる組み合わせステップと、

前記組み合わせステップの処理により組み合わせられた前記符号化パラメータの履歴を表す符号化履歴情報を生成する生成ステップと、

前記生成ステップの処理により生成された前記符号化履歴情報を、前記符号化ストリームに重乗して前記メディアに伝送する伝送ステップとを含む処理を実行させるコンピュータが読み取り可能なプログラムを提供することを特徴とする提供媒体。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、ストリーム伝送装置および方法、並びに提供媒体に関し、特に、動画像信

号を、例えば光磁気ディスクや磁気テープなどの記録媒体に記録し、これを再生して、ステレオ視が可能なディスプレイなどに表示したり、テレビ会議システム、テレビ電話システム、放送用機器など、動画像信号を伝送路を介して送信側から受信側に伝送し、受信側において、これを受信して表示する場合などに用いて好適なストリーム伝送装置および方法、並びに提供媒体に関する。

【0002】

【従来の技術】例えば、テレビ会議システム、テレビ電話システムなどのように、動画像信号を遠隔地に伝送するシステムにおいては、伝送路を効率良く利用するため、映像信号のライン相関やフレーム間相関を利用して、画像信号が圧縮符号化される。

【0003】画像信号が圧縮符号化される場合、生成されるビットストリームが、所定のビットレートになるように符号化が行われる。しかしながら、実運用上において、伝送路の都合により、ビットストリームのビットレートを変換する必要があることがある。このような場合、図 1 に示すようなトランスコード 131 により、符号化されている情報を一旦復号し、ビットレートが所定の値になるように、再び符号化する方法が一般的である。図 1 の例の場合、10Mbps で送られてきたビットストリームが、復号装置 132 により復号され、デジタルビデオ信号として符号化装置 133 に供給され、符号化装置 133 により、ビットレートが 5Mbps であるビットストリームに符号化されて出力される。

【0004】

【発明が解決しようとする課題】このように映像信号を再符号化する場合、符号化装置 133 には、図 2 に示すように、映像信号のライン相関やフレーム間相関を検出する動き検出部 134 が符号化部 135 の前段に必要となり、符号化装置 133 の規模が大きくなる課題があった。

【0005】また、例えば放送局においては、映像の編集が秒単位で行われるので、フレームの画像情報が他のフレームの画像情報と独立しているほうがよい。そこで、図 3 に示すように、低いビットレート（3 乃至 9Mbps）で転送しても画質が劣化しないように、相関関係を有するフレームの集合である GOP (Group of Picture) を構成するフレーム数が多い Long GOP の符号化装置 133-1 から出力されたビットストリームは、復号装置 132-1、符号化装置 133-2、復号装置 132-2、符号化装置 133-3 などを有する放送局の復号装置 132-1 と符号化装置 133-2 により、GOP を構成するフレーム数が少ない Short GOP に変換されて、高ビットレート（18 乃至 50Mbps）で伝送され、編集終了後、復号装置 132-2 と符号化装置 133-3 により、再度 Long GOP に変換されて、後段の復号装置 132-3 に出力される。このように、画像情報に符号化、復号が繰り返されると、符号化の度に使用される符号化パ

ラメータが変化するので画像情報が劣化する課題があった。

【0006】本発明はこのような状況に鑑みてなされたものであり、過去に演算した動きベクトルを用いて再符号化を行うことにより、装置の規模を小さくするとともに、再符号化に伴う画像の劣化を抑制することを、少ないデータ量で可能とするものである。

【0007】

【課題を解決するための手段】請求項1に記載のストリーム伝送装置は、過去の符号化処理において使用された複数の符号化パラメータから所定の符号化パラメータを選択的に組み合わせる組み合わせ手段と、組み合わせ手段により組み合わせられた符号化パラメータの履歴を表す符号化履歴情報を生成する生成手段と、生成手段により生成された符号化履歴情報を、符号化ストリームに重乗してメディアに伝送する伝送手段とを含むことを特徴とする。

【0008】請求項4に記載のストリーム伝送方法は、過去の符号化処理において使用された複数の符号化パラメータから所定の符号化パラメータを選択的に組み合わせる組み合わせステップと、組み合わせステップの処理により組み合わせられた符号化パラメータの履歴を表す符号化履歴情報を生成する生成ステップと、生成ステップの処理により生成された符号化履歴情報を、符号化ストリームに重乗してメディアに伝送する伝送ステップとを含むことを特徴とする。

【0009】請求項5に記載の提供媒体は、複数の符号化パラメータから所定のメディアを介して符号化ストリームを伝送するストリーム伝送装置に、過去の符号化処理において使用された複数の符号化パラメータから所定の符号化パラメータを選択的に組み合わせる組み合わせステップと、組み合わせステップの処理により組み合わせられた符号化パラメータの履歴を表す符号化履歴情報を生成する生成ステップと、生成ステップの処理により生成された符号化履歴情報を、符号化ストリームに重乗してメディアに伝送する伝送ステップとを含む処理を実行させるコンピュータが読み取り可能なプログラムを提供することを特徴とする提供媒体。

【0010】請求項1に記載のストリーム伝送装置、請求項4に記載のストリーム伝送方法、および請求項5に記載の提供媒体においては、過去の符号化処理において使用された複数の符号化パラメータから所定の符号化パラメータが選択的に組み合わせられ、組み合わせられた符号化パラメータの符号化履歴情報が生成され、符号化ストリームに重乗され、伝送される。

【0011】

【発明の実施の形態】本発明を適用したトランスコーダについて説明する前に、動画像信号の圧縮符号化について説明する。なお、本明細書においてシステムの用語は、複数の装置、手段などにより構成される全体的な装

置を意味するものである。

【0012】例えば、テレビ会議システム、テレビ電話システムなどのように、動画像信号を遠隔地に伝送するシステムにおいては、伝送路を効率良く利用するため、映像信号のライン相関やフレーム間相関を利用して、画像信号を圧縮符号化するようになされている。

【0013】ライン相関を利用すると、画像信号を、例えばDCT（離散コサイン変換）処理するなどして圧縮することができる。

【0014】また、フレーム間相関を利用すると、画像信号をさらに圧縮して符号化することが可能となる。例えば図4に示すように、時刻 t_1 乃至 t_3 において、フレーム画像PC1乃至PC3がそれぞれ発生している場合、フレーム画像PC1およびPC2の画像信号の差を演算して、PC12を生成し、また、フレーム画像PC2およびPC3の差を演算して、PC23を生成する。通常、時間的に隣接するフレームの画像は、それ程大きな変化を有していないため、両者の差を演算すると、その差分信号は小さな値のものとなる。そこで、この差分信号を符号化すれば、符号量を圧縮することができる。

【0015】しかしながら、差分信号のみを伝送したのでは、元の画像を復元することができない。そこで、各フレームの画像を、Iピクチャ、PピクチャまたはBピクチャの3種類のピクチャタイプのいずれかとし、画像信号を圧縮符号化するようにしている。

【0016】すなわち、例えば図5に示すように、フレームF1乃至F17までの17フレームの画像信号をグループオブピクチャ(GOP)とし、処理の1単位とする。そして、その先頭のフレームF1の画像信号はIピクチャとして符号化し、第2番目のフレームF2はBピクチャとして、また第3番目のフレームF3はPピクチャとして、それぞれ処理する。以下、第4番目以降のフレームF4乃至F17は、BピクチャまたはPピクチャとして交互に処理する。

【0017】Iピクチャの画像信号としては、その1フレーム分の画像信号をそのまま伝送する。これに対して、Pピクチャの画像信号としては、基本的には、図5に示すように、それより時間的に先行するIピクチャまたはPピクチャの画像信号からの差分を伝送する。さらにBピクチャの画像信号としては、基本的には、図6に示すように、時間的に先行するフレームまたは後行するフレームの両方の平均値からの差分を求め、その差分を符号化する。

【0018】図7は、このようにして、動画像信号を符号化する方法の原理を示している。同図に示すように、最初のフレームF1は、Iピクチャとして処理されるため、そのまま伝送データF1Xとして伝送路に伝送される（画像内符号化）。これに対して、第2のフレームF2は、Bピクチャとして処理されるため、時間的に先行するフレームF1と、時間的に後行するフレームF3の

平均値との差分が演算され、その差分が伝送データ F 2 X として伝送される。

【0019】ただし、この B ピクチャとしての処理は、さらに細かく説明すると、4 種類存在する。その第 1 の処理は、元のフレーム F 2 のデータをそのまま伝送データ F 2 X として伝送するものであり (SP 1) (イントラ符号化)、I ピクチャにおける場合と同様の処理となる。第 2 の処理は、時間的に後のフレーム F 3 からの差分を演算し、その差分 (SP 2) を伝送するものである (後方予測符号化)。第 3 の処理は、時間的に先行するフレーム F 1 との差分 (SP 3) を伝送するものである (前方予測符号化)。さらに第 4 の処理は、時間的に先行するフレーム F 1 と後行するフレーム F 3 の平均値との差分 (SP 4) を生成し、これを伝送データ F 2 X として伝送するものである (両方向予測符号化)。

【0020】実際には、上述した 4 つの方法のうちの伝送データが最も少なくなる方法が採用される。

【0021】なお、差分データを伝送するとき、差分を演算する対象となるフレームの画像 (参照画像) との間の動きベクトル x 1 (フレーム F 1 と F 2 の間の動きベクトル) (前方予測の場合)、もしくは x 2 (フレーム F 3 と F 2 の間の動きベクトル) (後方予測の場合)、または x 1 と x 2 の両方 (両方向予測の場合) が、差分データとともに伝送される。

【0022】また、P ピクチャのフレーム F 3 は、時間的に先行するフレーム F 1 を予測画像として、このフレームとの差分信号 (SP 3) と、動きベクトル x 3 が演算され、これが伝送データ F 3 X として伝送される (前方予測符号化)。あるいはまた、元のフレーム F 3 のデータが、そのままデータ F 3 X として伝送される (SP 1) (イントラ符号化)。これらの方法のうち、B ピクチャにおける場合と同様に、伝送データがより少なくなる方法が選択される。

【0023】図 8 は、上述した原理に基づいて、動画画像信号を符号化して伝送し、これを復号化する装置の構成例を示している。符号化装置 1 は、入力された映像信号を符号化し、伝送路としての記録媒体 3 に伝送するようになされている。そして、復号装置 2 は、記録媒体 3 に記録された信号を再生し、これを復号して出力するようになされている。

【0024】符号化装置 1 においては、入力された映像信号が前処理回路 11 に入力され、そこで輝度信号と色信号 (本実施の形態の場合、色差信号) が分離され、それぞれ A/D 変換器 12, 13 でアナログ信号からデジタル信号に変換される。A/D 変換器 12, 13 によりデジタル信号に変換された映像信号は、フレームメモリ 14 に供給され、記憶される。フレームメモリ 14 は、輝度信号を輝度信号フレームメモリ 15 に、また、色差信号を色差信号フレームメモリ 16 に、それぞれ記憶させる。

【0025】フォーマット変換回路 17 は、フレームメモリ 14 に記憶されたフレームフォーマットの信号を、ブロックフォーマットの信号に変換する。すなわち、図 9 に示すように、フレームメモリ 14 に記憶された映像信号は、1 ライン当り H ドットのラインが V ライン集められた、図 9 (A) に示すようなフレームフォーマットのデータとされている。フォーマット変換回路 17 は、この 1 フレームの信号を、図 9 (B) に示すように、16 ラインを単位として M 個のスライスに区分する。そして、各スライスは、M 個のマクロブロックに分割される。マクロブロックは、図 9 (C) に示すように、16 × 16 個の画素 (ドット) に対応する輝度信号により構成され、この輝度信号は、さらに 8 × 8 ドットを単位とするブロック Y [1] 乃至 Y [4] に区分される。そして、この 16 × 16 ドットの輝度信号には、8 × 8 ドットの C b 信号と、8 × 8 ドットの C r 信号が対応される。

【0026】このように、ブロックフォーマットに変換されたデータは、フォーマット変換回路 17 からエンコーダ 18 に供給され、ここでエンコード (符号化) が行われる。その詳細については、図 10 を参照して後述する。

【0027】エンコーダ 18 によりエンコードされた信号は、ビットストリームとして伝送路に出力される。例えば記録回路 19 に供給され、デジタル信号として記録媒体 3 に記録される。

【0028】再生回路 30 により記録媒体 3 より再生されたデータは、復号装置 2 のデコーダ 31 に供給され、デコードされる。デコーダ 31 の詳細については、図 15 を参照して後述する。

【0029】デコーダ 31 によりデコードされたデータは、フォーマット変換回路 32 に入力され、ブロックフォーマットからフレームフォーマットに変換される。そして、フレームフォーマットの輝度信号は、フレームメモリ 33 の輝度信号フレームメモリ 34 に供給されて記憶され、色差信号は色差信号フレームメモリ 35 に供給されて記憶される。輝度信号フレームメモリ 34 と色差信号フレームメモリ 35 から読み出された輝度信号と色差信号は、それぞれ D/A 変換器 36, 37 によりアナログ信号に変換され、後処理回路 38 に供給される。後処理回路 38 は、輝度信号と色差信号を合成して出力する。

【0030】次に図 10 を参照して、エンコーダ 18 の構成について説明する。符号化される画像データは、マクロブロック単位で動きベクトル検出回路 50 に入力される。動きベクトル検出回路 50 は、予め設定されている所定のシーケンスに従って、各フレームの画像データを、I ピクチャ、P ピクチャ、または B ピクチャとして処理する。シーケンシャルに入力される各フレームの画像を、I, P、または B のいずれのピクチャとして処理するかは、予め定められている (例えば、図 5 と図 6 に

示したように、フレームF1乃至F17により構成されるグループオブピクチャが、I, B, P, B, P, ... B, Pとして処理される。

【0031】Iピクチャとして処理されるフレーム（例えば、フレームF1）の画像データは、動きベクトル検出回路50からフレームメモリ51の前方原画像部51aに転送、記憶され、Bピクチャとして処理されるフレーム（例えば、フレームF2）の画像データは、原画像部51bに転送、記憶され、Pピクチャとして処理されるフレーム（例えば、フレームF3）の画像データは、

後方原画像部51cに転送、記憶される。
【0032】また、次のタイミングにおいて、さらにBピクチャ（フレームF4）またはPピクチャ（フレームF5）として処理すべきフレームの画像が入力されたとき、それまで後方原画像部51cに記憶されていた最初のPピクチャ（フレームF3）の画像データが、前方原画像部51aに転送され、次のBピクチャ（フレームF4）の画像データが、原画像部51bに記憶（上書き）され、次のPピクチャ（フレームF5）の画像データが、後方原画像部51cに記憶（上書き）される。この

ような動作が順次繰り返される。
【0033】フレームメモリ51に記憶された各ピクチャの信号は、そこから読み出され、予測モード切り替え回路52において、フレーム予測モード処理、またはフィールド予測モード処理が行なわれる。

【0034】さらにまた、予測判定回路54の制御の下に、演算部53において、画像内予測、前方予測、後方予測、または両方向予測の演算が行なわれる。これらの処理のうち、いずれの処理を行なうかは、予測誤差信号（処理の対象とされている参照画像と、これに対する予測画像との差分）に対応して決定される。このため、動きベクトル検出回路50は、この判定に用いられる予測誤差信号の絶対値和（自乗和でもよい）を生成する。

【0035】ここで、予測モード切り替え回路52におけるフレーム予測モードとフィールド予測モードについて説明する。

【0036】フレーム予測モードが設定された場合においては、予測モード切り替え回路52は、動きベクトル検出回路50より供給される4個の輝度ブロックY

[1]乃至Y[4]を、そのまま後段の演算部53に出力する。すなわち、この場合においては、図11に示すように、各輝度ブロックに奇数フィールドのラインのデータと、偶数フィールドのラインのデータとが混在した状態となっている。このフレーム予測モードにおいては、4個の輝度ブロック（マクロブロック）を単位として予測が行われ、4個の輝度ブロックに対して1個の動きベクトルが対応される。

【0037】これに対して、予測モード切り替え回路52は、フィールド予測モードにおいては、図11に示す構成で動きベクトル検出回路50より入力される信号

を、図12に示すように、4個の輝度ブロックのうち、輝度ブロックY[1]とY[2]を、例えば奇数フィールドのラインのドットだけで構成させ、他の2個の輝度ブロックY[3]とY[4]を、偶数フィールドのラインのドットだけで構成させて、演算部53に出力する。この場合においては、2個の輝度ブロックY[1]とY[2]に対して、1個の動きベクトルが対応され、他の2個の輝度ブロックY[3]とY[4]に対して、他の1個の動きベクトルが対応される。

【0038】動きベクトル検出回路50は、フレーム予測モードにおける予測誤差の絶対値和、およびフィールド予測モードにおける予測誤差の絶対値和を予測モード切り替え回路52に出力する。予測モード切り替え回路52は、フレーム予測モードとフィールド予測モードにおける予測誤差の絶対値和を比較し、その値が小さい予測モードに対応する処理を施して、データを演算部53に出力する。

【0039】ただし、このような処理は、実際には動きベクトル検出回路50で行われる。すなわち、動きベクトル検出回路50は、決定されたモードに対応する構成の信号を予測モード切り替え回路52に出力し、予測モード切り替え回路52は、その信号を、そのまま後段の演算部53に出力する。

【0040】なお、色差信号は、フレーム予測モードの場合、図11に示すように、奇数フィールドのラインのデータと偶数フィールドのラインのデータとが混在する状態で、演算部53に供給される。また、フィールド予測モードの場合、図12に示すように、各色差ブロックCb, Crの上半分（4ライン）が、輝度ブロックY[1], Y[2]に対応する奇数フィールドの色差信号とされ、下半分（4ライン）が、輝度ブロックY[3], Y[4]に対応する偶数フィールドの色差信号とされる。

【0041】また、動きベクトル検出回路50は、以下に示すようにして、予測判定回路54において、画像内予測、前方予測、後方予測、または両方向予測のいずれの予測を行なうかを決定するための予測誤差の絶対値和を生成する。

【0042】すなわち、画像内予測の予測誤差の絶対値和として、参照画像のマクロブロックの信号 A_{ij} の総和 $\sum A_{ij}$ の絶対値 $|\sum A_{ij}|$ と、マクロブロックの信号 A_{ij} の絶対値 $|A_{ij}|$ の総和 $\sum |A_{ij}|$ の差を求める。また、前方予測の予測誤差の絶対値和として、参照画像のマクロブロックの信号 A_{ij} と、予測画像のマクロブロックの信号 B_{ij} の差 $A_{ij}-B_{ij}$ の絶対値 $|A_{ij}-B_{ij}|$ の総和 $\sum |A_{ij}-B_{ij}|$ を求める。また、後方予測と両方向予測の予測誤差の絶対値和も、前方予測における場合と同様に（その予測画像を前方予測における場合と異なる予測画像に変更して）求める。

【0043】これらの絶対値和は、予測判定回路54に

供給される。予測判定回路 54 は、前方予測、後方予測および両方向予測の予測誤差の絶対値和のうちの最も小さいものを、インタ予測の予測誤差の絶対値和として選択する。さらに、このインタ予測の予測誤差の絶対値和と、画像内予測の予測誤差の絶対値和とを比較し、その小さい方を選択し、この選択した絶対値和に対応するモードを予測モードとして選択する。すなわち、画像内予測の予測誤差の絶対値和の方が小さければ、画像内予測モードが設定される。インタ予測の予測誤差の絶対値和の方が小さければ、前方予測、後方予測または両方向予測モードのうちの対応する絶対値和が最も小さかったモードが設定される。

【0044】このように、動きベクトル検出回路 50 は、参照画像のマクロブロックの信号を、フレームまたはフィールド予測モードのうち、予測モード切り替え回路 52 により選択されたモードに対応する構成で、予測モード切り替え回路 52 を介して演算部 53 に供給するとともに、4 つの予測モードのうちの予測判定回路 54 により選択された予測モードに対応する予測画像と参照画像の間の動きベクトルを検出し、可変長符号化回路 58 と動き補償回路 64 に出力する。上述したように、この動きベクトルとしては、対応する予測誤差の絶対値和が最小となるものが選択される。

【0045】予測判定回路 54 は、動きベクトル検出回路 50 が前方原画像部 51a より 1 ピクチャの画像データを読み出しているとき、予測モードとして、フレームまたはフィールド（画像）内予測モード（動き補償を行わないモード）を設定し、演算部 53 のスイッチ 53d を接点 a 側に切り替える。これにより、1 ピクチャの画像データが DCT モード切り替え回路 55 に入力される。

【0046】DCT モード切り替え回路 55 は、図 13 または図 14 に示すように、4 個の輝度ブロックのデータを、奇数フィールドのラインと偶数フィールドのラインが混在する状態（フレーム DCT モード）、または、分離された状態（フィールド DCT モード）、のいずれかの状態にして、DCT 回路 56 に出力する。

【0047】すなわち、DCT モード切り替え回路 55 は、奇数フィールドと偶数フィールドのデータを混在して DCT 処理した場合における符号化効率と、分離した状態において DCT 処理した場合の符号化効率とを比較し、符号化効率の良好なモードを選択する。

【0048】例えば、入力された信号を、図 13 に示すように、奇数フィールドと偶数フィールドのラインが混在する構成とし、上下に隣接する奇数フィールドのラインの信号と偶数フィールドのラインの信号の差を演算し、さらにその絶対値和（または自乗和）を求める。

【0049】また、入力された信号を、図 14 に示すように、奇数フィールドと偶数フィールドのラインが分離した構成とし、上下に隣接する奇数フィールドのライン同士の信号の差と、偶数フィールドのライン同士の信号

の差を演算し、それぞれの絶対値和（または自乗和）を求める。

【0050】さらに、両者（絶対値和）を比較し、小さい値に対応する DCT モードを設定する。すなわち、前者の方が小さければ、フレーム DCT モードを設定し、後者の方が小さければ、フィールド DCT モードを設定する。

【0051】そして、選択した DCT モードに対応する構成のデータを DCT 回路 56 に出力するとともに、選択した DCT モードを示す DCT フラグを、可変長符号化回路 58、および動き補償回路 64 に出力する。

【0052】予測モード切り替え回路 52 における予測モード（図 11 と図 12）と、この DCT モード切り替え回路 55 における DCT モード（図 13 と図 14）を比較して明らかなように、輝度ブロックに関しては、両者の各モードにおけるデータ構造は実質的に同一である。

【0053】予測モード切り替え回路 52 において、フレーム予測モード（奇数ラインと偶数ラインが混在するモード）が選択された場合、DCT モード切り替え回路 55 においても、フレーム DCT モード（奇数ラインと偶数ラインが混在するモード）が選択される可能性が高く、また予測モード切り替え回路 52 において、フィールド予測モード（奇数フィールドと偶数フィールドのデータが分離されたモード）が選択された場合、DCT モード切り替え回路 55 において、フィールド DCT モード（奇数フィールドと偶数フィールドのデータが分離されたモード）が選択される可能性が高い。

【0054】しかしながら、必ずしも常にこのようにモードが選択されるわけではなく、予測モード切り替え回路 52 においては、予測誤差の絶対値和が小さくなるようにモードが決定され、DCT モード切り替え回路 55 においては、符号化効率が良好となるようにモードが決定される。

【0055】DCT モード切り替え回路 55 より出力された 1 ピクチャの画像データは、DCT 回路 56 に入力されて DCT 処理され、DCT 係数に変換される。この DCT 係数は、量子化回路 57 に入力され、送信バッファ 59 のデータ蓄積量（バッファ蓄積量）に対応した量子化スケールで量子化された後、可変長符号化回路 58 に入力される。

【0056】可変長符号化回路 58 は、量子化回路 57 より供給される量子化スケール（スケール）に対応して、量子化回路 57 より供給される画像データ（いまの場合、1 ピクチャのデータ）を、例えばハフマン符号などの可変長符号に変換し、送信バッファ 59 に出力する。

【0057】可変長符号化回路 58 にはまた、量子化回路 57 より量子化スケール（スケール）、予測判定回路 54 より予測モード（画像内予測、前方予測、後方予測、または両方向予測のいずれが設定されたかを示すモード）、動きベクトル検出回路 50 より動きベクトル、

予測モード切り替え回路 52 より予測フラグ（フレーム予測モードまたはフィールド予測モードのいずれが設定されたかを示すフラグ）、および DCT モード切り替え回路 55 が出力する DCT フラグ（フレーム DCT モードまたはフィールド DCT モードのいずれが設定されたかを示すフラグ）が入力されており、これらも可変長符号化される。

【0058】送信バッファ 59 は、入力されたデータを一時蓄積し、蓄積量に対応するデータを量子化回路 57 に出力する。送信バッファ 59 は、そのデータ残量が許容上限値まで増量すると、量子化制御信号によって量子化回路 57 の量子化スケールを大きくすることにより、量子化データのデータ量を低下させる。また、これとは逆に、データ残量が許容下限値まで減少すると、送信バッファ 59 は、量子化制御信号によって量子化回路 57 の量子化スケールを小さくすることにより、量子化データのデータ量を増大させる。このようにして、送信バッファ 59 のオーバーフローまたはアンダフローが防止される。

【0059】そして、送信バッファ 59 に蓄積されたデータは、所定のタイミングで読み出され、伝送路に出力され、例えば記録回路 19 を介して記録媒体 3 に記録される。

【0060】一方、量子化回路 57 より出力された 1 ピクチャのデータは、逆量子化回路 60 に入力され、量子化回路 57 より供給される量子化スケールに対応して逆量子化される。逆量子化回路 60 の出力は、IDCT（逆離散コサイン変換）回路 61 に入力され、逆離散コサイン変換処理された後、演算器 62 を介してフレームメモリ 63 の前方予測画像部 63a 供給されて記憶される。

【0061】動きベクトル検出回路 50 は、シーケンシャルに入力される各フレームの画像データを、たとえば、I、B、P、B、P、B・・・のピクチャとしてそれぞれ処理する場合、最初に入力されたフレームの画像データを 1 ピクチャとして処理した後、次に入力されたフレームの画像データを B ピクチャとして処理する前に、さらにその次に入力されたフレームの画像データを P ピクチャとして処理する。B ピクチャは、後方予測を伴うため、後方予測画像としての P ピクチャが先に用意されていないと、復号することができないからである。

【0062】そこで動きベクトル検出回路 50 は、1 ピクチャの処理の次に、後方原画像部 51c に記憶されている P ピクチャの画像データの処理を開始する。そして、上述した場合と同様に、マクロブロック単位でのフレーム間差分（予測誤差）の絶対値和が、動きベクトル検出回路 50 から予測モード切り替え回路 52 と予測判定回路 54 に供給される。予測モード切り替え回路 52 と予測判定回路 54 は、この P ピクチャのマクロブロックの予測誤差の絶対値和に対応して、フレーム／フィールド予測モード、または画像内予測、前方予測、後方予

測、もしくは両方向予測の予測モードを設定する。

【0063】演算部 53 は、画像内予測モードが設定されたとき、スイッチ 53d を上述したように接点 a 側に切り替える。したがって、このデータは、1 ピクチャのデータと同様に、DCT モード切り替え回路 55、DCT 回路 56、量子化回路 57、可変長符号化回路 58、および送信バッファ 59 を介して伝送路に伝送される。また、このデータは、逆量子化回路 60、IDCT 回路 61、および演算器 62 を介してフレームメモリ 63 の後方予測画像部 63b に供給されて記憶される。

【0064】また、前方予測モードが設定された場合、スイッチ 53d が接点 b に切り替えられるとともに、フレームメモリ 63 の前方予測画像部 63a に記憶されている画像（いまの場合、1 ピクチャの画像）データが読み出され、動き補償回路 64 により、動きベクトル検出回路 50 が出力する動きベクトルに対応して動き補償される。すなわち、動き補償回路 64 は、予測判定回路 54 より前方予測モードの設定が指令されたとき、前方予測画像部 63a の読み出しアドレスを、動きベクトル検出回路 50 が、現在、出力しているマクロブロックの位置に対応する位置から動きベクトルに対応する分だけずらしてデータを読み出し、予測画像データを生成する。

【0065】動き補償回路 64 より出力された予測画像データは、演算器 53a に供給される。演算器 53a は、予測モード切り替え回路 52 より供給された参照画像のマクロブロックのデータから、動き補償回路 65 より供給された、このマクロブロックに対応する予測画像データを減算し、その差分（予測誤差）を出力する。この差分データは、DCT モード切り替え回路 55、DCT 回路 56、量子化回路 57、可変長符号化回路 58、および送信バッファ 59 を介して伝送路に伝送される。また、この差分データは、逆量子化回路 60、および IDCT 回路 61 により局所的に復号され、演算器 62 に入力される。

【0066】この演算器 62 にはまた、演算器 53a に供給されている予測画像データと同一のデータが供給されている。演算器 62 は、IDCT 回路 61 が出力する差分データに、動き補償回路 64 が出力する予測画像データを加算する。これにより、元の（復号した）P ピクチャの画像データが得られる。この P ピクチャの画像データは、フレームメモリ 63 の後方予測画像部 63b に供給されて記憶される。

【0067】動きベクトル検出回路 50 は、このように、1 ピクチャと P ピクチャのデータが前方予測画像部 63a と後方予測画像部 63b にそれぞれ記憶された後、次に B ピクチャの処理を実行する。予測モード切り替え回路 52 と予測判定回路 54 は、マクロブロック単位でのフレーム間差分の絶対値和の大きさに対応して、フレーム／フィールドモードを設定し、また、予測モードを画像内予測モード、前方予測モード、後方予測モー

ド、または両方向予測モードのいずれかに設定する。

【0068】上述したように、画像内予測モードまたは前方予測モードの時、スイッチ53dは接点aまたはbに切り替えられる。このとき、Pピクチャにおける場合と同様の処理が行われ、データが伝送される。

【0069】これに対して、後方予測モードまたは両方向予測モードが設定された時、スイッチ53dは、接点cまたはdにそれぞれ切り替えられる。

【0070】スイッチ53dが接点cに切り替えられている後方予測モードの時、後方予測画像部63bに記憶されている画像（いまの場合、Pピクチャの画像）データが読み出され、動き補償回路64により、動きベクトル検出回路50が出力する動きベクトルに対応して動き補償される。すなわち、動き補償回路64は、予測判定回路54より後方予測モードの設定が指令されたとき、後方予測画像部63bの読み出しアドレスを、動きベクトル検出回路50が、現在、出力しているマクロブロックの位置に対応する位置から動きベクトルに対応する分だけずらしてデータを読み出し、予測画像データを生成する。

【0071】動き補償回路64より出力された予測画像データは、演算器53bに供給される。演算器53bは、予測モード切り替え回路52より供給された参照画像のマクロブロックのデータから、動き補償回路64より供給された予測画像データを減算し、その差分を出力する。この差分データは、DCTモード切り替え回路55、DCT回路56、量子化回路57、可変長符号化回路58、および送信バッファ59を介して伝送路に伝送される。

【0072】スイッチ53dが接点dに切り替えられている両方向予測モードの時、前方予測画像部63aに記憶されている画像（いまの場合、Iピクチャの画像）データと、後方予測画像部63bに記憶されている画像（いまの場合、Pピクチャの画像）データが読み出され、動き補償回路64により、動きベクトル検出回路50が出力する動きベクトルに対応して動き補償される。

【0073】すなわち、動き補償回路64は、予測判定回路54より両方向予測モードの設定が指令されたとき、前方予測画像部63aと後方予測画像部63bの読み出しアドレスを、動きベクトル検出回路50がいま出力しているマクロブロックの位置に対応する位置から動きベクトル（この場合の動きベクトルは、前方予測画像用と後方予測画像用の2つとなる）に対応する分だけずらしてデータを読み出し、予測画像データを生成する。

【0074】動き補償回路64より出力された予測画像データは、演算器53cに供給される。演算器53cは、動きベクトル検出回路50より供給された参照画像のマクロブロックのデータから、動き補償回路64より供給された予測画像データの平均値を減算し、その差分を出力する。この差分データは、DCTモード切り替え回

路55、DCT回路56、量子化回路57、可変長符号化回路58、および送信バッファ59を介して伝送路に伝送される。

【0075】Bピクチャの画像は、他の画像の予測画像とされることがないため、フレームメモリ63には記憶されない。

【0076】なお、フレームメモリ63において、前方予測画像部63aと後方予測画像部63bは、必要に応じてバンク切り替えが行われ、所定の参照画像に対して、一方または他方に記憶されているものを、前方予測画像あるいは後方予測画像として切り替えて出力することができる。

【0077】上述した説明においては、輝度ブロックを中心として説明をしたが、色差ブロックについても同様に、図11乃至図14に示すマクロブロックを単位として処理されて伝送される。なお、色差ブロックを処理する場合の動きベクトルは、対応する輝度ブロックの動きベクトルを垂直方向と水平方向に、それぞれ1/2にしたものが用いられる。

【0078】図15は、図8のデコーダ31の構成を示すブロック図である。伝送路（記録媒体3）を介して伝送された符号化された画像データは、図示せぬ受信回路で受信されたり、再生装置で再生され、受信バッファ81に一時記憶された後、復号回路90の可変長復号化回路82に供給される。可変長復号化回路82は、受信バッファ81より供給されたデータを可変長復号化し、動きベクトル、予測モード、予測フラグ、およびDCTフラグを動き補償回路87に出力し、量子化スケールを逆量子化回路83に出力するとともに、復号された画像データを逆量子化回路83に出力する。

【0079】逆量子化回路83は、可変長復号化回路82より供給された画像データを、同じく可変長復号化回路82より供給された量子化スケールに従って逆量子化し、IDCT回路84に出力する。逆量子化回路83より出力されたデータ（DCT係数）は、IDCT回路84により、逆離散コサイン変換処理が施され、演算器85に供給される。

【0080】IDCT回路84より演算器85に供給された画像データが、Iピクチャのデータである場合、そのデータは演算器85より出力され、演算器85に後に入力される画像データ（PまたはBピクチャのデータ）の予測画像データ生成のために、フレームメモリ86の前方予測画像部86aに供給されて記憶される。また、このデータは、フォーマット変換回路32（図8）に出力される。

【0081】IDCT回路84より供給された画像データが、その1フレーム前の画像データを予測画像データとするPピクチャのデータであり、前方予測モードのデータである場合、フレームメモリ86の前方予測画像部86aに記憶されている、1フレーム前の画像データ（I

ピクチャのデータ)が読み出され、動き補償回路87で可変長復号化回路82より出力された動きベクトルに対応する動き補償が施される。そして、演算器85において、IDCT回路84より供給された画像データ(差分のデータ)と加算され、出力される。この加算されたデータ、すなわち、復号されたPピクチャのデータは、演算器85に後に入力される画像データ(BピクチャまたはPピクチャのデータ)の予測画像データ生成のために、フレームメモリ86の後方予測画像部86bに供給されて記憶される。

【0082】Pピクチャのデータであっても、画像内予測モードのデータは、Iピクチャのデータと同様に、演算器85において処理は行われず、そのまま後方予測画像部86bに記憶される。

【0083】このPピクチャは、次のBピクチャの次に表示されるべき画像であるため、この時点では、まだフォーマット変換回路32へ出力されない(上述したように、Bピクチャの後に入力されたPピクチャが、Bピクチャより先に処理され、伝送されている)。

【0084】IDCT回路84より供給された画像データが、Bピクチャのデータである場合、可変長復号化回路82より供給された予測モードに対応して、フレームメモリ86の前方予測画像部86aに記憶されているIピクチャの画像データ(前方予測モードの場合)、後方予測画像部86bに記憶されているPピクチャの画像データ(後方予測モードの場合)、または、その両方の画像データ(両方向予測モードの場合)が読み出され、動き補償回路87において、可変長復号化回路82より出力された動きベクトルに対応する動き補償が施されて、予測画像が生成される。但し、動き補償を必要としない場合(画像内予測モードの場合)、予測画像は生成されない。

【0085】このようにして、動き補償回路87で動き補償が施されたデータは、演算器85において、IDCT回路84の出力と加算される。この加算出力は、フォーマット変換回路32に出力される。

【0086】ただし、この加算出力はBピクチャのデータであり、他の画像の予測画像生成のために利用されることがないため、フレームメモリ86には記憶されない。

【0087】Bピクチャの画像が出力された後、後方予測画像部86bに記憶されているPピクチャの画像データが読み出され、動き補償回路87を介して演算器85に供給される。但し、このとき、動き補償は行われない。

【0088】なお、このデコーダ31には、図8のエンコーダ18における予測モード切り替え回路52とDCTモード切り替え回路55に対応する回路が図示されていないが、これらの回路に対応する処理、すなわち、奇数フィールドと偶数フィールドのラインの信号が分離され

た構成を元の構成に必要に応じて戻す処理は、動き補償回路87により実行される。

【0089】また、上述した説明においては、輝度信号の処理について説明したが、色差信号の処理も同様に行われる。ただし、この場合の動きベクトルは、輝度信号用の動きベクトルを、垂直方向および水平方向に1/2にしたものが用いられる。

【0090】図16は、符号化された画像の品質を示している。画像の品質(SNR:Signal to Noise Ratio)は、ピクチャタイプに対応して制御され、Iピクチャ、およびPピクチャは高品質とされ、Bピクチャは、I、Pピクチャに比べて劣る品質とされて伝送される。これは、人間の視覚特性を利用した手法であり、全ての画像品質を平均化するよりも、品質を振動させたほうが視覚上の画質が良くなるためである。このピクチャタイプに対応した画質の制御は、図10の量子化回路57により実行される。

【0091】図17は、本発明を適用したトランスコーダ101の構成を示しており、図18は、そのさらに詳細な構成を示している。復号装置102は、所定のビットレート(この例の場合、10Mbps)のビットストリームに含まれる(多重化されている)符号化された画像信号を、ビットストリームに含まれる(多重化されている)そのビットストリームの現符号化パラメータ(フレーム/フィールドDCTフラグ、フレーム/フィールド予測フラグ、予測モード、ピクチャタイプ、動きベクトル、マクロブロック情報、および量子化スケール)を用いて復号し、符号化パラメータ多重装置103に出力するとともに、現符号化パラメータも符号化パラメータ多重装置103に出力するようになされている。

【0092】復号装置102はまた、ビットストリームに含まれるユーザデータを復号、分離し、履歴復号装置104に出力する。その詳細は後述するが、このユーザデータには、直近の3世代分の符号化パラメータで構成される世代履歴情報が含まれている。これに対して、現符号化パラメータは、例えばgroup_of_pictures_header(1), extension_and_user_data(1), picture_header(), picture_coding_extension(), extensions_data(2), picture_data(),または、sequence_extension()に含まれている(後述する図41)。履歴復号装置104は、入力されたユーザデータを復号し、3世代分の符号化パラメータを含む世代履歴情報を符号化パラメータ多重装置103に出力する。

【0093】なお、復号装置102は、図8の復号装置2のデコーダ31(図15)を図19に示すデコーダ111に変更したものである。デコーダ111の可変長復号化回路112は、現符号化パラメータをビットストリームから抽出し、所定の回路に供給するとともに、世代履歴情報を含むユーザデータを抽出し、履歴復号装置104に出力するようになされている。デコーダ111の

その他の構成は、デコーダ31と同様であるので、その説明は省略する。

【0094】符号化パラメータ多重装置103は、復号された画像データの空き領域（その詳細は、図21を参照して説明する）に4世代分の符号化パラメータを書き込み（多重化し）、ベースバンドのデジタルビデオ信号として、粗結合された（符号化パラメータ伝送用の専用バス等が設けられていない）符号化パラメータ分離装置105に出力する。符号化パラメータ分離装置105は、ベースバンドのデジタルビデオ信号から、画像データと、符号化装置106で符号化に用いる符号化パラメータを分離して符号化装置106に供給するようになっている。

【0095】符号化パラメータ分離装置105はまた、入力されたベースバンドのデジタルビデオ信号から、符号化装置106で用いる符号化パラメータを除く3世代分の符号化パラメータを抽出し、履歴符号化装置107に出力する。履歴符号化装置107は、入力された3世代分の符号化パラメータをユーザデータに書き込み、そのユーザデータを符号化装置106に出力する。

【0096】符号パラメータが書き込まれる画像データのフォーマットについて、図20と図21を参照して説明する。1個のマクロブロックは、図20に示すように、 $16 \times 16 (= 256)$ 画素で構成される。この 16×16 画素のデータは、 8×8 画素の輝度信号の4個のブロックと、 8×8 画素の色差信号の4個のブロック（2個のCbブロックと2個のCrブロック）から構成されている。図21は、このうちの、4画素分の輝度データ $Y[0][x]$ 乃至 $Y[3][x]$ 、2画素分の色差データ $Cr[0][x]$ 、 $Cr[1][x]$ 、および2画素分の色差データ $Cb[0][x]$ 、 $Cb[1][x]$ （ $x=0$ 乃至9）を伝送するフォーマットを表している。従って、図20に示す256画素のマクロブロックのデータは、図21に示すフォーマットを64個（ $= 256 / 4$ ）用意することで伝送される。

【0097】実際の1画素分の輝度データ、および1画素分の色差データは、いずれも8ビットのデータとされている。しかしながら、図21の画像データの伝送フォーマットとしては、1画素あたり、10ビット分の容量（D0乃至D9）が設けられているので、2画素分の領域（D0、D1）が不要となる。図21のフォーマット全体では16ビット（ $= 2 \times 8$ ）分の空き容量が存在するので、1マクロブロックでは、1024ビット（ $16 \text{ ビット} \times 64 \text{ 個}$ ）の情報が記録できるので、この1024ビット分の領域に本来の画像データ以外の符号化パラメータを書き込む。なお、1個のマクロブロックに対応する符号化パラメータは、256ビットの情報量があるので、この領域には、過去4回の符号化に使用された符号化パラメータを記録することができる。

【0098】このように、符号化パラメータ多重装置103から符号化パラメータ分離装置105に伝送される

画像データ（デジタルビデオ信号）には、輝度信号Y、色差信号Cr、Cbを記載する領域として、10画素分（D0乃至D9）の領域が設けられている。しかしながら実際に輝度信号Y等が書き込まれる領域は、D2乃至D9の8画素分の領域であり、D0、D1の領域は利用されない。そこで、この2ビットの領域を符号化パラメータの書き込み用領域として利用する。これにより、図20の 16×16 画素の所定の位置の画素の下位2ビットに、符号化パラメータが書き込まれることとなる。

10 【0099】符号化パラメータ多重装置103において、符号化パラメータが書き込まれる画像データのフォーマットについて、図20と図21を参照して説明する。1個のマクロブロックは、図20に示すように、 16×16 画素で構成される。この 16×16 画素のデータは、 8×8 画素の輝度信号 $Y[0][x]$ 乃至 $Y[3][x]$ と、 8×8 画素の色差信号 $Cr[0][x]$ 、 $Cr[1][x]$ および $Cb[0][x]$ 、 $Cb[1][x]$ （ $x=2$ 乃至9）から構成されている。例えば、輝度信号 $Y[0][9]$ は、 8×8 画素の1行目の画素（8画素）の輝度信号を示している。1画素当たりの輝度信号の情報量は8ビットなので、輝度信号 $Y[0][9]$ の情報量は、8（画素） \times 8（ビット） $= 64$ ビットとなる。色差信号についても同様である。

20 【0100】これに対して、画像データのフォーマットは、図21に示すように、10行分の領域（D0乃至D9）が設けられているので、2行分の領域（D0、D1）が不要となる。この空き領域には、 $64 \text{ ビット} \times 16 = 1024$ ビットの情報が記録できるので、この2行分の領域に本来の画像データ以外の符号化パラメータが書き込まれる。なお、1個のマクロブロックに対応する符号化パラメータは、256ビットの情報量があるので、この領域には、過去4回の符号化に使用された符号化パラメータを記録することができる。

30 【0101】符号化パラメータ多重装置103から符号化パラメータ分離装置105に伝送される画像データ（デジタルビデオ信号）には、輝度信号Y、色差信号Cr、Cbを記載する領域として、10行分（D0乃至D9）の領域が設けられている。しかしながら実際に輝度信号Y等が書き込まれる領域は、D2乃至D9の8行分の領域であり、D0、D1の領域は利用されない。そこで、この2行の領域を符号化パラメータの書き込み用領域として利用する。これにより、図20の 16×16 画素の所定の位置の画素の下位2ビットに、符号化パラメータが書き込まれることとなる。

40 【0102】符号化装置106は、これから行う符号化のための符号化パラメータとして供給された現符号化パラメータを利用して画像データを符号化するとともに、履歴符号化装置107から供給されるユーザデータをビットストリームに多重化して、所定のビットレート（この例の場合、5Mbps）でSDTI(Serial Data Transfer Interface)108-i（ $i=1, 2, \dots, N$ ）（後述

する図33)に出力するようになされている。

【0103】なお、符号化装置106は、図8の符号化装置1のエンコーダ18(図10)を図22に示すエンコーダ121に変更したものである。エンコーダ121は、エンコーダ18から符号化パラメータを生成する動きベクトル検出回路50、フレームメモリ51、予測モード切り替え回路52、予測判定回路54、およびDCTモード切り替え回路55を削除し、履歴符号化装置107の出力するユーザデータを可変長符号化回路58で可変長符号化するようにしたものである。エンコーダ121のその他の構成は、エンコーダ18と同様であるので、その説明は省略する。

【0104】次に、図18における履歴復号装置104と履歴符号化装置107についてさらに説明する。同図に示すように、履歴復号装置104は、復号装置102より供給されるユーザデータをデコードするユーザデータデコーダ201、ユーザデータデコーダ201の出力を変換するコンバータ202、およびコンバータ202の出力から履歴情報を再生するヒストリデコーダ203により構成されている。

【0105】また、履歴符号化装置107は、符号化パラメータ分離装置105より供給される3世代分の符号化パラメータをフォーマット化するヒストリフォーマッタ211、ヒストリフォーマッタ211の出力を変換するコンバータ212、コンバータ212の出力をユーザデータのフォーマットにフォーマットするユーザデータフォーマッタ213により構成されている。

【0106】ユーザデータデコーダ201は、復号装置102より供給されるユーザデータをデコードして、コンバータ202に出力する。詳細は図54を参照して後述するが、ユーザデータ(user_data())は、user_data_start_codeとuser_dataからなり、MPEG規格においてはuser_dataの中に、連続する23ビットの"0"(start_codeと同一のコード)を発生させることを禁止している。これは、そのデータが、start_codeとして誤検出されるのを防止するためである。履歴情報(history_stream())は、ユーザデータエリアに(MPEG規格のuser_dataの一種として)記述され、その中には、このような連続する23ビット以上の"0"が存在することがあり得るので、これを、連続する23ビット以上の"0"が発生しないように処理して、converted_history_stream() (後述する図41)に変換する必要がある。この変換を行うのは、履歴符号化装置107のコンバータ212である。履歴復号装置104のコンバータ202は、このコンバータ212と逆の変換処理を行う(連続する23ビット以上の"0"を発生させないために挿入された"1"を除去する)ものである。

【0107】ヒストリデコーダ203は、コンバータ202の出力から履歴情報を生成し、符号化パラメータ多重装置103に出力する。

【0108】一方、履歴符号化装置107においては、ヒストリフォーマッタ211が符号化パラメータ分離装置105より供給される3世代分の符号化パラメータを履歴情報のフォーマットに変換する。このフォーマットには、固定長のもの(後述する図43乃至図49)と、可変長のもの(後述する図50)とがある。これらの詳細については後述する。

【0109】ヒストリフォーマッタ211により、フォーマット化された履歴情報は、コンバータ212において、converted_history_stream()に変換される。これは、上述したように、user_data()のstart_codeが誤検出されないようにするためのものである。すなわち、履歴情報内には連続する23ビット以上の"0"が存在するが、user_data中には連続する23ビット以上の"0"を配置することができないので、この禁止項目に触れないようにコンバータ212によりデータを変換するのである。

【0110】ユーザデータフォーマッタ213は、コンバータ212より供給されるconverted_history_stream()に、後述する図41に基づいて、History_Data_IDを付加し、さらに、user_data_stream_codeを付加して、video stream中に挿入できるMPEG規格のuser_dataを生成し、符号化装置106に出力する。

【0111】図23は、ヒストリフォーマッタ211の構成例を表している。その符号語変換器301と符号長変換器305には、符号化パラメータ(今回、履歴情報として伝送する符号化パラメータ)(項目データ)と、この符号化パラメータを配置するストリームを特定する情報(例えば、シンタックスの名称(例えば、後述するsequence_headerの名称))(項目NO.)が、符号化パラメータ分離装置105から供給されている。符号語変換器301は、入力された符号化パラメータを、指示されたシンタックスに対応する符号語に変換し、バレルシフタ302に出力する。バレルシフタ302は、符号語変換器301より入力された符号語を、アドレス発生回路306より供給されるシフト量に対応する分だけシフトし、バイト単位の符号語として、スイッチ303に出力する。アドレス発生回路306が出力するビットセレクト信号により切り換えられるスイッチ303は、ビット分設けられており、バレルシフタ302より供給される符号語を、RAM304に供給し、記憶させる。このときの書き込みアドレスは、アドレス発生回路306から指定される。また、アドレス発生回路306から読み出しアドレスが指定されたとき、RAM304に記憶されているデータ(符号語)が読み出され、後段のコンバータ212に供給されるとともに、必要に応じて、スイッチ303を介してRAM304に再び供給され、記憶される。

【0112】符号長変換器305は、入力されるシンタックスと符号化パラメータとから、その符号化パラメータの符号長を決定し、アドレス発生回路306に出力す

る。アドレス発生回路 306 は、入力された符号長に対応して、上述したシフト量、ビットセレクト信号、書き込みアドレス、または読み出しアドレスを生成し、それらを、それぞれパレルシフタ 302、スイッチ 303、または RAM 304 に供給する。

【0113】以上のように、ヒストリフォーマット 211 は、いわゆる可変長符号化器として構成され、入力された符号化パラメータを可変長符号化して出力する。

【0114】図 24 は、以上のようにしてヒストリフォーマット化されたデータをデコードするヒストリデコーダ 203 の構成例を表している。このヒストリデコーダ 203 には、コンバータ 202 から供給された符号化パラメータのデータが RAM 311 に供給されて、記憶される。このときの書き込みアドレスは、アドレス発生回路 315 から供給される。アドレス発生回路 315 はまた、所定のタイミングで読み出しアドレスを発生し、RAM 311 に供給する。このとき、RAM 311 は、読み出しアドレスに記憶されているデータを読み出し、パレルシフタ 312 に出力する。パレルシフタ 312 は、アドレス発生回路 315 が出力するシフト量に対応する分だけ、入力されるデータをシフトし、逆符号長変換器 313 と逆符号語変換器 314 に出力する。

【0115】逆符号長変換器 313 にはまた、コンバータ 202 から、符号化パラメータが配置されているストリームのシンタックスの名称（項目 NO.）が供給されている。逆符号長変換器 313 は、そのシンタックスに基づいて、入力されたデータ（符号語）から符号長を求め、求めた符号長をアドレス発生回路 315 に出力する。

【0116】また、逆符号語変換器 314 は、パレルシフタ 312 より供給されたデータを、シンタックスに基づいて復号し（逆符号語化し）、符号化パラメータ多重装置 103 に出力する。

【0117】また、逆符号語変換器 314 は、どのような符号語が含まれているのかを特定するのに必要な情報（符号語の区切りを決定するのに必要な情報）を抽出し、アドレス発生回路 315 に出力する。アドレス発生回路 315 は、この情報と逆符号長変換器 313 より入力された符号長に基づいて、書き込みアドレスおよび読み出しアドレスを発生し、RAM 311 に出力するとともに、シフト量を発生し、パレルシフタ 312 に出力する。

【0118】図 25 は、コンバータ 212 の構成例を表している。この例においては、ヒストリフォーマット 211 とコンバータ 212 の間に配置されているバッファメモリ 320 の、コントローラ 326 が出力する読み出しアドレスから 8 ビットのデータが読み出され、D 型フリップフロップ（D-FF）321 に供給され、保持されるようになされている。そして、D 型フリップフロップ 321 より読み出されたデータは、スタッフ回路 32

3 に供給されるとともに、8 ビットの D 型フリップフロップ 322 にも供給され、保持される。D 型フリップフロップ 322 より読み出された 8 ビットのデータは、D 型フリップフロップ 321 より読み出された 8 ビットのデータと合成され、16 ビットのパレルデータとして、スタッフ回路 323 に供給される。

【0119】スタッフ回路 323 は、コントローラ 326 より供給されるスタッフ位置を示す信号（stuff position）の位置に符号“1”を挿入し（スタッフイングし）、合計 17 ビットのデータとして、パレルシフタ 324 に出力する。

【0120】パレルシフタ 324 は、コントローラ 326 より供給されるシフト量を示す信号（shift）に基づいて入力されたデータをシフトして、8 ビットのデータを抽出し、8 ビットの D 型フリップフロップ 325 に出力する。D 型フリップフロップ 325 に保持されたデータは、そこから読み出され、バッファメモリ 327 を介して、後段のユーザデータフォーマット 213 に供給される。この時、コントローラ 326 は、出力するデータとともに、書き込みアドレスを発生し、コンバータ 212 とユーザデータフォーマット 213 との間に介在するバッファメモリ 327 に供給する。

【0121】図 26 は、スタッフ回路 323 の構成例を表している。D 型フリップフロップ 322、321 より入力された 16 ビットのデータは、それぞれスイッチ 331-16 乃至 331-1 の接点 a に入力されている。スイッチ 331-i（i=0 乃至 15）の接点 c には、MSB 側（図中上方）に隣接するスイッチのデータが供給されている。例えば、スイッチ 331-12 の接点 c には、MSB 側に隣接するスイッチ 331-13 の接点 a に供給されている LSB から 13 番目のデータが供給されており、スイッチ 331-13 の接点 c には、MSB 側に隣接するスイッチ 331-14 の接点 a に供給されている LSB 側から 14 番目のデータが供給されている。

【0122】但し、LSB に対応するスイッチ 331-1 よりさらに下側のスイッチ 331-0 の接点 a は、開放されている。また、MSB に対応するスイッチ 331-16 の接点 c は、それより上位のスイッチが存在しないため、開放されている。

【0123】各スイッチ 331-0 乃至 331-16 の接点 b には、データ“1”が供給されている。

【0124】デコーダ 332 は、コントローラ 326 より供給されるデータ“1”を挿入する位置を示す信号 stuff position に対応して、スイッチ 331-0 乃至 331-16 のうち、1 つのスイッチを接点 b 側に切り替え、それより LSB 側のスイッチは、接点 c 側にそれぞれ切り替えさせ、それより MSB 側のスイッチは、接点 a 側に切り替えさせる。

【0125】図 26 は、LSB 側から 13 番目にデータ“1”を挿入する場合の例を示している。従って、この場

合、スイッチ 331-0 乃至スイッチ 331-12 は、いずれも接点 c 側に切り替えられ、スイッチ 331-13 は、接点 b 側に切り替えられ、スイッチ 331-14 乃至スイッチ 331-16 は、接点 a 側に切り替えられている。

【0126】図 25 のコンバータ 212 は、以上のような構成により、22 ビットの符号を 23 ビットに変換して、出力することになる。

【0127】図 27 は、図 25 のコンバータ 212 の各部の出力データのタイミングを表している。コンバータ 212 のコントローラ 326 がバイト単位のクロックに同期して、読み出しアドレス（図 27 (A)）を発生すると、バッファメモリ 320 から、それに対応するデータが、バイト単位で読み出され、D 型フリップフロップ 321 に一旦保持される。そして、D 型フリップフロップ 321 より読み出されたデータ（図 27 (B)）は、スタッフ回路 323 に供給されるとともに、D 型フリップフロップ 322 に供給され、保持される。D 型フリップフロップ 322 に保持されたデータは、そこからさらに読み出され（図 27 (C)）、スタッフ回路 323 に供給される。

【0128】従って、スタッフ回路 323 の入力（図 27 (D)）は、読み出しアドレス A1 のタイミングにおいて、最初の 1 バイトのデータ D0 とされ、次の読み出しアドレス A2 のタイミングにおいて、1 バイトのデータ D0 と 1 バイトのデータ D1 より構成される 2 バイトのデータとなり、さらに読み出しアドレス A3 のタイミングにおいては、データ D1 とデータ D2 より構成される 2 バイトのデータとなる。

【0129】スタッフ回路 323 には、データ"1"を挿入する位置を示す信号 stuff position（図 27 (E)）がコントローラ 326 より供給される。スタッフ回路 323 のデコーダ 332 は、スイッチ 331-16 乃至 331-0 のうち、この信号 stuff position に対応するスイッチを接点 b に切り換え、それより LSB 側のスイッチを接点 c 側に切り換え、さらにそれより MSB 側のスイッチを接点 a 側に切り換える。これにより、データ"1"が挿入されるので、スタッフ回路 323 からは、信号 stuff position で示す位置に、データ"1"が挿入されたデータ（図 27 (F)）が出力される。

【0130】パレルシフト 324 は、入力されたデータを、コントローラ 326 より供給される信号 shift（図 27 (G)）で示される量だけパレルシフトして、出力する（図 27 (H)）。この出力がさらに D 型フリップフロップ 325 で一旦保持された後、後段に出力される（図 27 (I)）。

【0131】D 型フリップフロップ 325 より出力されるデータには、22 ビットのデータの次に、データ"1"が挿入されている。従って、データ"1"と、次のデータ"1"の間には、その間のビットが全て 0 であつ

たとしても、0 のデータの連続する数は 22 となる。

【0132】図 28 は、コンバータ 202 の構成例を表している。このコンバータ 202 の D 型フリップフロップ 341 乃至コントローラ 346 よりなる構成は、図 25 に示したコンバータ 212 の D 型フリップフロップ 321 乃至コントローラ 326 と基本的に同様の構成であるが、コンバータ 212 におけるスタッフ回路 323 に代えて、ディリット回路 343 が挿入されている点がコンバータ 212 における場合と異なっている。その他の構成は、図 25 のコンバータ 212 における場合と同様である。

【0133】すなわち、このコンバータ 202 においては、コントローラ 346 が出力する削除するビットの位置を示す信号 delete position に従って、ディリット回路 343 が、そのビット（図 25 のスタッフ回路 323 で挿入されたデータ"1"）が削除される。

【0134】その他の動作は、図 25 のコンバータ 212 における場合と同様である。

【0135】図 29 は、ディリット回路 343 の構成例を表している。この構成例においては、D 型フリップフロップ 342、341 より入力された 16 ビットのデータのうち、LSB 側の 15 ビットが、それぞれ対応するスイッチ 351-0 乃至 351-14 の接点 a に供給されている。各スイッチの接点 b には、1 ビットだけ MSB 側のデータが供給されている。デコーダ 352 は、コントローラ 346 より供給される信号 delete position により指定されるビットを削除して、15 ビットのデータとして出力するようになされている。

【0136】図 29 は、LSB から第 13 番目のビットがディリットされる状態を示している。従って、この場合、スイッチ 351-0 乃至スイッチ 351-11 が接点 a 側に切り替えられ、LSB から第 12 番目までの 12 ビットが、そのまま選択、出力されている。また、スイッチ 351-12 乃至 351-14 は、それぞれ接点 b 側に切り替えられているので、第 14 番目乃至第 16 番目のデータが、第 13 番目乃至第 15 番目のビットのデータとして選択、出力される。

【0137】図 26 のスタッフ回路 323 および図 29 のディリット回路 343 の入力が 16 ビットとなっているのは、それぞれ図 25 のコンバータ 212 のスタッフ回路 323 の入力が、D 型フリップフロップ 322、321 より供給される 16 ビットとされており、また、図 28 のコンバータ 202 においても、ディリット回路 343 の入力が、D 型フリップフロップ 342、341 により 16 ビットとされているためである。図 25 において、スタッフ回路 323 の出力する 17 ビットをパレルシフト 324 でパレルシフトすることにより、例えば 8 ビットを最終的に選択、出力しているのと同様に、図 28 のコンバータ 202 においても、ディリット回路 343 の出力する 15 ビットのデータを、パレルシフト 344

4で所定量だけバレルシフトすることにより、8ビットのデータとしている。

【0138】図30は、コンバータ212の他の構成例を表している。この構成例においては、カウンタ361が入力データのうち、連続する0のビットの数をカウントし、そのカウント結果をコントローラ326に出力するようになされている。コントローラ326は、例えばカウンタ361が連続する0のビットを22個カウントしたとき、信号stuff positionをスタッフ回路323に出力する。また、このとき、コントローラ326は、カウンタ361をリセットし、再び連続する0のビットの数をカウンタ361にカウントさせる。

【0139】その他の構成と動作は、図25における場合と同様である。

【0140】図31は、コンバータ202の他の構成例を表している。この構成例においては、入力データのうち、連続する0の数をカウンタ371がカウントし、そのカウント結果をコントローラ346に出力するようになされている。カウンタ371のカウント値が22に達したとき、コントローラ346は、信号delete positionをディリット回路343に出力するとともに、カウンタ371をリセットし、再び新たな連続する0のビットの数をカウンタ371にカウントさせる。その他の構成は、図28における場合と同様である。

【0141】このように、この構成例においては、所定のパターン（データ”0”の連続する数）に基づいて、マーカービットとしてのデータ”1”が挿入され、また、削除されることになる。

【0142】図30と図31に示す構成は、図25と図28に示す構成よりも効率的な処理が可能となる。但し、変換後の長さが元の履歴情報に依存することになる。

【0143】図32は、ユーザデータフォーマット213の構成例を表している。この例においては、コントローラ383がコンバータ212とユーザデータフォーマット213との間に配置されているバッファメモリ（図示せず）に読み出しアドレスを出力すると、そこから読み出されたデータが、ユーザデータフォーマット213のスイッチ382の接点a側に供給される。ROM381には、ユーザデータスタートコード、データIDなどのuser_data()を生成するのに必要なデータが記憶されている。コントローラ313は、所定のタイミングにおいて、スイッチ382を接点a側または接点b側に切り替え、ROM381に記憶されているデータ、またはコンバータ212より供給されるデータを適宜選択し、出力する。これにより、user_data()のフォーマットのデータが符号化装置106に出力される。

【0144】なお、図示は省略するが、ユーザデータコード201は、図32のROM381より読み出され、挿入されたデータを削除するスイッチを介して、入力デ

ータを出力するようにすることで実現することができる。

【0145】図33は、例えば映像編集スタジオにおいて、複数のトランスコード101-1乃至101-Nが直列に接続されて使用される状態を示している。各トランスコード101-i（i=1乃至N）の符号化パラメータ多重装置103-iは、上述した符号化パラメータ用の領域の最も古い符号化パラメータが記録されている区画に、自己が用いた最新の符号化パラメータを上書きする。このことにより、ベースバンドの画像データには、同一のマクロブロックに対応する直近の4世代分の符号化パラメータ（世代履歴情報）が記録されることになる。

【0146】各符号化装置106-iのエンコード121-i（図22）は、その可変長符号化回路58において、符号化パラメータ分離装置105-iから供給される今回用いる符号化パラメータに基づいて、量子化回路57より供給されるビデオデータを符号化する。このようにして生成されるビットストリーム（例えば、picture_header()）中に、その現符号化パラメータは多重化される。

【0147】可変長符号化回路58はまた、履歴符号化装置107-iより供給されるユーザデータ（世代履歴情報を含む）を、出力するビットストリーム中に多重化する（図21に示すような埋め込み処理ではなく、ビットストリーム中に多重化される）。そして、符号化装置106-iの出力するビットストリームは、SDTI108-iを介して、後段のトランスコード101-(i+1)に輸入される。

【0148】トランスコード101-iとトランスコード101-(i+1)は、それぞれ図18に示すように構成されている。従って、その処理は、図18を参照して説明した場合と同様となる。

【0149】実際の符号化パラメータの履歴を利用した符号化として、現在Iピクチャとして符号化されていたものを、PもしくはBピクチャに変更したい場合、過去の符号化パラメータの履歴を見て、過去にPもしくはBピクチャであった場合を探し、これらの履歴が存在した場合は、その動きベクトルなどのパラメータを利用して、ピクチャタイプを変更する。反対に過去に履歴がない場合は、動き検出を行わないピクチャタイプの変更を断念する。もちろん履歴がない場合であっても、動き検出を行えばピクチャタイプを変更できる。

【0150】図21に示すフォーマットの場合、4世代分の符号化パラメータを埋め込むようにしたが、I、P、Bの各ピクチャタイプのパラメータを埋め込むようにすることもできる。図34は、この場合のフォーマットの例を示している。この例では、同一のマクロブロックが、過去にピクチャタイプの変更を伴って符号化されたときにおける、ピクチャタイプ毎に1世代分の符号化

パラメータ（ピクチャ履歴情報）が記録される。したがって、図 19 に示したデコーダ 111、および図 22 に示したエンコーダ 121 は、現在（最新）、1 世代前、2 世代前、および 3 世代前の符号化パラメータの代わりに、I ピクチャ、P ピクチャ、および B ピクチャに対応する 1 世代分の符号化パラメータを入出力することになる。

【0151】また、この例の場合、Cb[1][x]とCr[1][x]の領域は利用しないので、Cb[1][x]とCr[1][x]の領域を有さない 4:2:0 フォーマットの画像データにも本発明を適用することができる。

【0152】この例の場合、復号装置 102 は、符号化パラメータを復号と同時に取り出し、ピクチャタイプを判定して、画像信号のピクチャタイプに対応した場所に符号化パラメータを書き込んで（多重化して）符号化パラメータ分離装置 105 に出力する。符号化パラメータ分離装置 105 は、符号化パラメータを分離し、これから符号化したいピクチャタイプと、入力された過去の符号化パラメータを考慮して、ピクチャタイプを変更しながら再符号化を行うことができる。

【0153】次に、各トランスコーダ 101 において、変更が可能なピクチャタイプを判定する処理について、図 35 のフローチャートを参照して説明する。なお、この処理はトランスコーダ 101 におけるピクチャタイプの変更は、過去の動きベクトルを利用するので、動き検出を行わないで実行されることを前提としている。また、以下に説明する処理は、符号化パラメータ分離装置 105 により実行される。

【0154】ステップ S1 において、ピクチャタイプ毎に 1 世代分の符号化パラメータ（ピクチャ履歴情報）が符号化パラメータコントローラ 122 に入力される。

【0155】ステップ S2 において、符号化パラメータ分離装置 105 は、ピクチャ履歴情報に B ピクチャに変更したときの符号化パラメータが存在するか否かを判定する。ピクチャ履歴情報に B ピクチャに変更したときの符号化パラメータが存在すると判定された場合、ステップ S3 に進む。

【0156】ステップ S3 において、符号化パラメータ分離装置 105 は、ピクチャ履歴情報に P ピクチャに変更したときの符号化パラメータが存在するか否かを判定する。ピクチャ履歴情報に P ピクチャに変更したときの符号化パラメータが存在すると判定された場合、ステップ S4 に進む。

【0157】ステップ S4 において、符号化パラメータ分離装置 105 は、変更可能なピクチャタイプが I ピクチャ、P ピクチャ、および B ピクチャであると判断する。

【0158】ステップ S3 において、ピクチャ履歴情報に P ピクチャに変更したときの符号化パラメータが存在しないと判定された場合、ステップ S5 に進む。

【0159】ステップ S5 において、符号化パラメータ分離装置 105 は、変更可能なピクチャタイプが I ピクチャ、および B ピクチャであると判断する。さらに、符号化パラメータ分離装置 105 は、特殊処理（B ピクチャの履歴情報に含まれる後方予測ベクトルを使わず、前方予測ベクトルだけを使う）を施すことにより、擬似的に P ピクチャに変更可能であると判断する。

【0160】ステップ S2 において、ピクチャ履歴情報に B ピクチャに変更したときの符号化パラメータが存在しないと判定された場合、ステップ S6 に進む。

【0161】ステップ S6 において、符号化パラメータ分離装置 105 は、ピクチャ履歴情報に P ピクチャに変更したときの符号化パラメータが存在するか否かを判定する。ピクチャ履歴情報に P ピクチャに変更したときの符号化パラメータが存在すると判定された場合、ステップ S7 に進む。

【0162】ステップ S7 において、符号化パラメータ分離装置 105 は、変更可能なピクチャタイプが I ピクチャ、および P ピクチャであると判断する。さらに、符号化パラメータ分離装置 105 は、特殊処理（P ピクチャに履歴情報に含まれる前方予測ベクトルだけを使う）を施すことにより、B ピクチャに変更可能であると判断する。

【0163】ステップ S6 において、ピクチャ履歴情報に P ピクチャに変更したときの符号化パラメータが存在しないと判定された場合、ステップ S8 に進む。ステップ S8 において、符号化パラメータ分離装置 105 は、動きベクトルが存在しないので、変更可能なピクチャタイプが I ピクチャだけである（I ピクチャなので I ピクチャ以外には変更できない）と判断する。

【0164】ステップ S4、S5、S7、S8 の処理の次にステップ S9 において、符号化パラメータ分離装置 105 は、変更可能なピクチャタイプを表示装置（図示せず）に表示してユーザに通知する。

【0165】図 36 は、ピクチャタイプ変更の例を示している。ピクチャタイプを変更する場合、GOP を構成するフレーム数を変更される。すなわち、この例の場合、N=15（GOP のフレーム数 N=15）、M=3（GOP 内の I、または P ピクチャの出現周期 M=3）のフレームから構成される Long GOP（第 1 世代）から、N=1、M=1 のフレームで構成される Short GOP（第 2 世代）に変換され、再度、N=15、M=3 のフレームから構成される Long GOP（第 3 世代）に変換されている。なお、図中において破線は、GOP の境界を示している。

【0166】第 1 世代から第 2 世代にピクチャタイプが変更される場合において、上述した変更可能ピクチャタイプ判定処理の説明から明らかなように、全てのフレームは、ピクチャタイプを I ピクチャに変更することが可能である。このピクチャタイプ変更のとき、動画像（第 0 世代）が第 1 世代に変換されたときに演算された全て

の動きベクトルは、ピクチャ履歴情報に保存された（残された）状態となる。次に、再度Long GOPに変換される（第2世代から第3世代にピクチャタイプが変更される）場合、第0世代から第1世代に変換されたときのピクチャタイプ毎の動きベクトルが保存されているので、これを再利用することにより、画質劣化を抑えて、再度、Long GOPに変換することが可能となる。

【0167】図37は、ピクチャタイプ変更の他の例を示している。この例の場合、N=14、M=2であるLong GOP（第1世代）から、N=2、M=2であるShort GOP（第2世代）に変換され、さらに、N=1、M=1であるフレーム数が1のShort GOP（第3世代）に変換されて、フレーム数NがランダムなGOP（第4世代）に変換される。

【0168】この例においても、第0世代から第1世代に変換されたときのピクチャタイプ毎の動きベクトルが、第3世代から第4世代への変換のときまで保存される。そこで、図37に示すように、複雑にピクチャタイプを変更しても、保存されている符号化パラメータを再利用されることにより、画質劣化を小さく抑えることができる。さらに、保存されている符号化パラメータの量子化スケールを有効に利用すれば画質劣化の少ない符号化を実現できる。

【0169】この量子化スケールの再利用について、図38を参照して説明する。図38は、所定のフレームが、第1世代から第4世代まで常に、Iピクチャに変換されており、ビットレートだけが、4Mbps、18Mbps、または50Mbpsに変更されていることを示している。

【0170】例えば、第1世代(4Mbps)から第2世代(18Mbps)への変換の際に、ビットレートの高速化に伴って、細かい量子化スケールで再符号化しても画質は向上しない。なぜならば、過去において粗い量子化ステップで量子化されたデータは、復元しないからである。したがって、図38に示すように、途中でビットレートが高速化しても、それに伴って細かい量子化ステップで量子化することは、情報量が増加するだけであって画質の向上には繋がらない。したがって、過去のもっとも粗い（大きい）量子化スケールを維持するように制御すれば、最も無駄が無く、効率的な符号化が可能となる。

【0171】上述したように、ビットレートが変更されるときは、過去の量子化スケールの履歴を利用して符号化することは非常に有効である。

【0172】この量子化制御処理について、図39のフローチャートを参照して説明する。ステップS11において、符号化パラメータ分離装置105は、入力されたピクチャ履歴情報に、いまから変換するピクチャタイプの符号化パラメータが存在するか否かを判定する。変換するピクチャタイプの符号化パラメータが存在すると判定された場合、ステップS12に進む。

【0173】ステップS12において、符号化パラメータ分離装置105は、ピクチャ履歴情報の対照となる符

号化パラメータから量子化スケール(Q_history)を抽出する。

【0174】ステップS13において、符号化パラメータ分離装置105は、送信バッファ59から量子化回路57にフィードバックされる量子化スケールの候補値Q_feedbackを読み取る。

【0175】ステップS14において、符号化パラメータ分離装置105は、Q_historyがQ_feedbackよりも大きい（粗い）か否かを判定する。Q_historyがQ_feedbackよりも大きいと判定された場合、ステップS15に進む。

【0176】ステップS15において、符号化パラメータ分離装置105は、量子化スケールとしてQ_historyを量子化回路57に出力する。量子化回路57は、Q_historyを用いて量子化を実行する。

【0177】ステップS16において、フレームに含まれる全てのマクロブロックが量子化されたか否かが判定される。全てのマクロブロックが量子化されていないと判定された場合、ステップS13に戻り、ステップS13乃至S16の処理が、全てのマクロブロックが量子化されるまで繰り返される。

【0178】ステップS14において、Q_historyがQ_feedbackよりも大きくない（細かい）いと判定された場合、ステップS17に進む。

【0179】ステップS17において、符号化パラメータ分離装置105は、量子化スケールとしてQ_feedbackを量子化回路57に出力する。量子化回路57は、Q_feedbackを用いて量子化を実行する。

【0180】ステップS11において、変換するピクチャタイプの符号化パラメータが存在しないと判定された場合、ステップS18に進む。

【0181】ステップS18において、量子化回路57は、送信バッファ59からフィードバックされる量子化スケールの候補値Q_feedbackを受け付ける。

【0182】ステップS19において、量子化回路57は、Q_feedbackを用いて量子化を実行する。

【0183】ステップS20において、フレームに含まれる全てのマクロブロックが量子化されたか否かが判定される。全てのマクロブロックが量子化されていないと判定された場合、ステップS18に戻り、ステップS18乃至S20の処理が、全てのマクロブロックが量子化されるまで繰り返される。

【0184】なお、本実施の形態におけるトランスコーダ101の内部においては、上述したように、復号側と符号側が粗結合されており、符号化パラメータを画像データに多重化させて伝送させたが、図40に示すように、復号装置102と符号化装置106を符号化パラメータ伝送用の高速バス111で接続する（密結合する）ようにしてもよい。

【0185】図41は、MPEGのビデオストリームをデコ

ードするためのシンタックスを表わした図である。デコードは、このシンタックスに従ってMPEGビットストリームをデコードすることによって、ビットストリームから意味のある複数のデータ項目（データエレメント）を抽出する。以下に説明するシンタックスは、図において、その関数や条件文は細活字で表わされ、そのデータエレメントは、太活字で表されている。データ項目は、その名称、ビット長、及びそのタイプと伝送順序を示すニーモニック（Mnemonic）で記述されている。

【0186】まず、この図41に示されているシンタックスにおいて使用されている関数について説明する。

【0187】next_start_code()関数は、ビットストリーム中に記述されているスタートコードを探すための関数である。この図41に示されたシンタックスにおいて、このnext_start_code()関数の次に、sequence_header()関数とsequence_extension()関数とが順に配置されているので、このビットストリームには、このsequence_header()関数とsequence_extension()関数によって定義されたデータエレメントが記述されている。従って、ビットストリームのデコード時には、このnext_start_code()関数によって、sequence_header()関数とsequence_extension()関数の先頭に記述されているスタートコード（データエレメントの一種）をビットストリーム中から見つけ、それを基準にして、sequence_header()関数とsequence_extension()関数をさらに見つけ、それらによって定義された各データエレメントをデコードする。

【0188】尚、sequence_header()関数は、MPEGビットストリームのシーケンス層のヘッダデータを定義するための関数であって、sequence_extension()関数は、MPEGビットストリームのシーケンス層の拡張データを定義するための関数である。

【0189】sequence_extension()関数の次に配置されているdo{ }while構文は、while文によって定義されている条件が真である間、do文の{ }内の関数に基いて記述されたデータエレメントをデータストリーム中から抽出するための構文である。すなわち、do{ }while構文によって、while文によって定義されている条件が真である間、ビットストリーム中から、do文内の関数に基いて記述されたデータエレメントを抽出するデコード処理が行われる。

【0190】このwhile文に使用されているnextbits()関数は、ビットストリーム中に現れるビット又はビット列と、次にデコードされるデータエレメントとを比較するための関数である。この図41のシンタックスの例では、nextbits()関数は、ビットストリーム中のビット列とビデオシーケンスの終わりを示すsequence_end_codeとを比較し、ビットストリーム中のビット列とsequence_end_codeとが一致しないときに、このwhile文の条件が真となる。従って、sequence_extension()関数の次に配置されているdo{ }while構文は、ビットストリーム中

に、ビデオシーケンスの終わりを示すsequence_end_codeが現れない間、do文中の関数によって定義されたデータエレメントがビットストリーム中に記述されていることを示している。

【0191】ビットストリーム中には、sequence_extension()関数によって定義された各データエレメントの次には、extension_and_user_data(0)関数によって定義されたデータエレメントが記述されている。このextension_and_user_data(0)関数は、MPEGビットストリームのシーケンス層の拡張データとユーザデータを定義するための関数である。

【0192】このextension_and_user_data(0)関数の次に配置されているdo{ }while構文は、while文によって定義されている条件が真である間、do文の{ }内の関数に基いて記述されたデータエレメントを、ビットストリーム中から抽出するための関数である。このwhile文において使用されているnextbits()関数は、ビットストリーム中に現れるビット又はビット列と、picture_start_code又はgroup_start_codeとの一致を判断するための関数であって、ビットストリーム中に現れるビット又はビット列と、picture_start_code又はgroup_start_codeとが一致する場合には、while文によって定義された条件が真となる。よって、このdo{ }while構文は、ビットストリーム中において、picture_start_code又はgroup_start_codeが現れた場合には、そのスタートコードの次に、do文中の関数によって定義されたデータエレメントのコードが記述されているので、このpicture_start_code又はgroup_start_codeによって示されるスタートコードを探し出すことによって、ビットストリーム中からdo文中に定義されたデータエレメントを抽出することができる。

【0193】このdo文の最初に記述されているif文は、ビットストリーム中にgroup_start_codeが現れた場合、という条件を示している。このif文による条件が真である場合には、ビットストリーム中には、このgroup_start_codeの次にgroup_of_picture_header(1)関数及びextension_and_user_data(1)関数によって定義されているデータエレメントが順に記述されている。

【0194】このgroup_of_picture_header(1)関数は、MPEGビットストリームのGOP層のヘッダデータを定義するための関数であって、extension_and_user_data(1)関数は、MPEGビットストリームのGOP層の拡張データ(extension_data)及びユーザデータ(user_data)を定義するための関数である。

【0195】さらに、このビットストリーム中には、group_of_picture_header(1)関数及びextension_and_user_data(1)関数によって定義されているデータエレメントの次に、picture_header()関数とpicture_coding_extension()関数によって定義されたデータエレメントが記述されている。もちろん、先に説明したif文の条件が真と

ならない場合には、group_of_picture_header(1)関数及びextension_and_user_data(1)関数によって定義されているデータエレメントは記述されていないので、extension_and_user_data(0)関数によって定義されているデータエレメントの次に、picture_header()関数とpicture_coding_extension()関数によって定義されたデータエレメントが記述されている。

【0196】このpicture_header()関数は、MPEGビットストリームのピクチャ層のヘッダデータを定義するための関数であって、picture_coding_extension()関数は、MPEGビットストリームのピクチャ層の第1の拡張データを定義するための関数である。

【0197】次のwhile文は、このwhile文によって定義されている条件が真である間、次のif文の条件判断を行うための関数である。このwhile文において使用されているnextbits()関数は、ビットストリーム中に現れるビット列と、extension_start_code又はuser_data_start_codeとの一致を判断するための関数であって、ビットストリーム中に現れるビット列と、extension_start_code又はuser_data_start_codeとが一致する場合には、このwhile文によって定義された条件が真となる。

【0198】第1のif文は、ビットストリーム中に現れるビット列とextension_start_codeとの一致を判断するための関数である。ビットストリーム中に現れるビット列と32ビットのextension_start_codeとが一致する場合には、ビットストリーム中において、extension_start_codeの次にextension_data(2)関数によって定義されるデータエレメントが記述されている。

【0199】第2のif文は、ビットストリーム中に現れるビット列とuser_data_start_codeとの一致を判断するための構文であって、ビットストリーム中に現れるビット列と32ビットのuser_data_start_codeとが一致する場合には、第3のif文の条件判断が行われる。このuser_data_start_codeは、MPEGビットストリームのピクチャ層のユーザデータエリアの開始を示すためのスタートコードである。

【0200】第3のif文は、ビットストリーム中に現れるビット列とHistory_Data_IDとの一致を判断するための構文である。ビットストリーム中に現れるビット列とこの32ビットのHistory_Data_IDとが一致する場合には、このMPEGビットストリームのピクチャ層のユーザデータエリアにおいて、この32ビットのHistory_Data_IDによって示されるコードの次に、converted_history_stream()関数によって定義されるデータエレメントが記述されている。

【0201】converted_history_stream()関数は、MPEG符号化時に使用したあらゆる符号化パラメータを伝送するための履歴情報及び履歴データを記述するための関数である。このconverted_history_stream()関数によって定義されているデータエレメントの詳細は、図43乃至

図50を参照して、history_stream()として後述する。また、このHistory_Data_IDは、MPEGビットストリームのピクチャ層のユーザデータエリアに記述されたこの履歴情報及び履歴データが記述されている先頭を示すためのスタートコードである。

【0202】else文は、第3のif文において、条件が非真であることを示すための構文である。従って、このMPEGビットストリームのピクチャ層のユーザデータエリアにおいて、converted_history_stream()関数によって定義されたデータエレメントが記述されていない場合には、user_data()関数によって定義されたデータエレメントが記述されている。

【0203】図41において、履歴情報は、converted_history_stream()に記述され、user_data()に記述される訳ではないが、このconverted_history_stream()は、MPEG規格のuser_dataの一種として記述されるので、本明細書中においては、場合によって、履歴情報がuser_dataに記述されるとも説明するが、それは、MPEG規格のuser_dataの一種として記述されるということを意味する。

【0204】picture_data()関数は、MPEGビットストリームのピクチャ層のユーザデータの次に、スライス層及びマクロブロック層に関するデータエレメントを記述するための関数である。通常は、このpicture_data()関数によって示されるデータエレメントは、ビットストリームのピクチャ層のユーザデータエリアに記述されたconverted_history_stream()関数によって定義されるデータエレメント又はuser_data()関数によって定義されたデータエレメントの次に記述されているが、ピクチャ層のデータエレメントを示すビットストリーム中に、extension_start_code又はuser_data_start_codeが存在しない場合には、このpicture_data()関数によって示されるデータエレメントは、picture_coding_extension()関数によって定義されるデータエレメントの次に記述されている。

【0205】このpicture_data()関数によって示されるデータエレメントの次に、sequence_header()関数とsequence_extension()関数とによって定義されたデータエレメントが順に配置されている。このsequence_header()関数とsequence_extension()関数によって記述されたデータエレメントは、ビデオストリームのシーケンスの先頭に記述されたsequence_header()関数とsequence_extension()関数によって記述されたデータエレメントと全く同じである。このように同じデータをストリーム中に記述する理由は、ビットストリーム受信装置側でデータストリームの途中(例えばピクチャ層に対応するビットストリーム部分)から受信が開始された場合に、シーケンス層のデータを受信できなくなり、ストリームをデコード出来なくなることを防止するためである。

【0206】この最後のsequence_header()関数とsequence

nce_extension()関数とによって定義されたデータエレメントの次、つまり、データストリームの最後には、シーケンスの終わりを示す32ビットのsequence_end_codeが記述されている。

【0207】以上のシンタックスの基本的な構成の概略を示すと、図42に示すようになる。

【0208】次に、converted_history_stream()関数によって定義されたヒストリーストリームに関して説明する。

【0209】このconverted_history_stream()は、MPEGのピクチャ層のユーザデータエリアに履歴情報を示すヒストリーストリームを挿入するための関数である。尚、「converted」の意味は、スタートエミュレーションを防止するために、ユーザエリアに挿入すべき履歴データから構成される履歴ストリームの少なくとも22ビット毎にマーカービット(1ビット)を挿入する変換処理を行ったストリームであることを意味している。

【0210】このconverted_history_stream()は、以下に説明する固定長の履歴ストリーム(図43乃至図49)又は可変長の履歴ストリーム(図50)のいずれかの形式で記述される。エンコード側において固定長の履歴ストリームを選択した場合には、デコード側において履歴ストリームから各データエレメントをデコードするための回路及びソフトウェアが簡単になるというメリットがある。一方、エンコード側において可変長の履歴ストリームを選択した場合には、エンコードにおいてピクチャ層のユーザエリアに記述される履歴情報(データエレメント)を必要に応じて任意に選択することができるので、履歴ストリームのデータ量を少なくすることができ、その結果、符号化されたビットストリーム全体のデータレートを低減することができる。

【0211】本発明において説明する「履歴情報」、「履歴データ」、「履歴パラメータ」とは、過去の符号化処理において使用した符号化パラメータ(又はデータエレメント)を意味し、現在の(最終段の)符号化処理において使用した符号化パラメータを意味するものではない。例えば、第1世代の符号化処理において、あるピクチャをIピクチャで符号化して伝送し、次なる第2世代の符号化処理において、このピクチャを今度はPピクチャとして符号化して伝送し、さらに、第3世代の符号化処理において、このピクチャをBピクチャで符号化して伝送する例をあげて説明する。第3世代の符号化処理において使用した符号化パラメータが、第3世代の符号化処理において生成された符号化ビットストリームのシーケンス層、GOP層、ピクチャ層、スライス層及びマクロブロック層の所定位置に記述されている。一方、過去の符号化処理である第1世代及び第2世代の符号化処理において使用した符号化パラメータは、第3世代の符号化処理において使用した符号化パラメータが記述されるシーケンス層やGOP層に記述されるのでは無く、既に説

明したシンタックスに従って、符号化パラメータの履歴情報として、ピクチャ層のユーザデータエリアに記述される。

【0212】まず、固定長の履歴ストリームシンタックスについて図43乃至図49を参照して説明する。

【0213】最終段(例えば第3世代)の符号化処理において生成されたビットストリームのピクチャ層のユーザデータエリアには、まず最初に、過去(例えば第1世代及び第2世代)の符号化処理において使用されていたシーケンス層のシーケンスヘッダに含まれる符号化パラメータが、履歴ストリームとして挿入される。尚、過去の符号化処理において生成されたビットストリームのシーケンス層のシーケンスヘッダ等の履歴情報は、最終段の符号化処理において生成されたビットストリームのシーケンス層のシーケンスヘッダに挿入されることは無いという点に注意すべきである。

【0214】過去の符号化処理で使用したシーケンスヘッダに含まれるデータエレメントは、sequence_header_code、sequence_header_present_flag、horizontal_size_value、vertical_size_value、aspect_ratio_information、frame_rate_code、bit_rate_value、marker_bit、VBV_buffer_size_value、constrained_parameter_flag、load_intra_quantizer_matrix、intra_quantizer_matrix、load_non_intra_quantizer_matrix、及びnon_intra_quantizer_matrix等から構成される。

【0215】sequence_header_codeは、シーケンス層のスタート同期コードを表すデータである。sequence_header_present_flagは、sequence_header内のデータが有効か無効かを示すデータである。horizontal_size_valueは、画像の水平方向の画素数の下位12ビットから成るデータである。vertical_size_valueは、画像の縦のライン数の下位12ビットからなるデータである。aspect_ratio_informationは、画素のアスペクト比(縦横比)または表示画面アスペクト比を表すデータである。frame_rate_codeは、画像の表示周期を表すデータである。

【0216】bit_rate_valueは、発生ビット量に対する制限のためのビット・レートの下位18ビット(400bsp単位で切り上げる)データである。marker_bitは、スタートコードエミュレーションを防止するために挿入されるビットデータである。VBV_buffer_size_valueは、発生符号量制御用の仮想バッファ(ビデオバッファペリフアイヤー)の大きさを決める値の下位10ビットデータである。constrained_parameter_flagは、各パラメータが制限以内であることを示すデータである。load_intra_quantizer_matrixは、イントラMB用量子化マトリックス・データの存在を示すデータである。intra_quantizer_matrixは、イントラMB用量子化マトリックスの値を示すデータである。load_non_intra_quantizer_matrixは、非イントラMB用量子化マトリックス・データの存在を示すデータである。non_intra_quantizer_matrixは、非

イントラMB用量子化マトリックスの値を表すデータである。

【0217】最終段の符号化処理において生成されたビットストリームのピクチャ層のユーザデータエリアには、過去の符号化処理において使用されたシーケンス層のシーケンスエクステンションを表わすデータエレメントが、履歴ストリームとして記述される。

【0218】この過去の符号化処理で使用したシーケンスエクステンションを表わすデータエレメントは、extension_start_code、extension_start_code_identifier、sequence_extension_present_flag、profile_and_level_indication、progressive_sequence、chroma_format、horizontal_size_extension、vertical_size_extension、bit_rate_extension、vbv_buffer_size_extension、low_delay、frame_rate_extension_n、及び frame_rate_extension_d等のデータエレメントである。

【0219】extension_start_codeは、エクステンションデータのスタート同期コードを表すデータである。extension_start_code_identifierは、どの拡張データが送られるかを示すデータである。sequence_extension_present_flagは、シーケンスエクステンション内のデータが有効であるか無効であることを示すデータである。profile_and_level_indicationは、ビデオデータのプロファイルとレベルを指定するためのデータである。progressive_sequenceは、ビデオデータが順次走査であることを示すデータである。chroma_formatは、ビデオデータの色差フォーマットを指定するためのデータである。

【0220】horizontal_size_extensionは、シーケンスヘッダのhorizontal_size_valueに加える上位2ビットのデータである。vertical_size_extensionは、シーケンスヘッダのvertical_size_valueに加える上位2ビットのデータである。bit_rate_extensionは、シーケンスヘッダのbit_rate_valueに加える上位12ビットのデータである。vbv_buffer_size_extensionは、シーケンスヘッダのvbv_buffer_size_valueに加える上位8ビットのデータである。low_delayは、Bピクチャを含まないことを示すデータである。frame_rate_extension_nは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを取得するためのデータである。frame_rate_extension_dは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを取得するためのデータである。

【0221】続いて、ビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたシーケンス層のシーケンスディスプレイエクステンションを表わすデータエレメントが、履歴ストリームとして記述される。

【0222】このシーケンスディスプレイエクステンションとして記述されているデータエレメントは、extension_start_code、extension_start_code_identifier、s

sequence_display_extension_present_flag、video_format、color_description、color_primaries、transfer_characteristics、matrix_coefficients、display_horizontal_size、及びdisplay_vertical_sizeから構成される。

【0223】extension_start_codeは、エクステンションデータのスタート同期コードを表すデータである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。sequence_display_extension_present_flagは、シーケンスディスプレイエクステンション内のデータエレメントが有効か無効かを示すデータである。video_formatは、原信号の映像フォーマットを表すデータである。color_descriptionは、色空間の詳細データがあることを示すデータである。color_primariesは、原信号の色特性の詳細を示すデータである。transfer_characteristicsは、光電変換がどのように行われたのかの詳細を示すデータである。matrix_coefficientsは、原信号が光の三原色からどのように変換されたかの詳細を示すデータである。display_horizontal_sizeは、意図するディスプレイの活性領域（水平サイズ）を表すデータである。display_vertical_sizeは、意図するディスプレイの活性領域（垂直サイズ）を表すデータである。

【0224】続いて、最終段の符号化処理において生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において生成されたマクロブロックの位相情報を示すマクロブロックアサインメントデータ（macroblock_assignment_in_user_data）が、履歴ストリームとして記述される。

【0225】このマクロブロックの位相情報を示すmacroblock_assignment_in_user_dataは、macroblock_assignment_present_flag、v_phase、h_phase等のデータエレメントから構成される。

【0226】このmacroblock_assignment_present_flagは、macroblock_assignment_in_user_data内のデータエレメントが有効か無効かを示すデータである。v_phaseは、画像データからマクロブロックを切り出す際の垂直方向の位相情報を示すデータである。h_phaseは、画像データからマクロブロックを切り出す際の水平方向の位相情報を示すデータである。

【0227】続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたGOP層のGOPヘッダを表わすデータエレメントが、履歴ストリームとして記述されている。

【0228】このGOPヘッダを表わすデータエレメントは、group_start_code、group_of_picture_header_present_flag、time_code、closed_gop、及びbroken_linkから構成される。

【0229】group_start_codeは、GOP層の開始同期コ

ードを示すデータである。group_of_picture_header_present_flagは、group_of_picture_header内のデータエレメントが有効であるか無効であるかを示すデータである。time_codeは、GOPの先頭ピクチャのシーケンスの先頭からの時間を示すタイムコードである。closed_gopは、GOP内の画像が他のGOPから独立再生可能なことを示すフラグデータである。broken_linkは、編集などのためにGOP内の先頭のBピクチャが正確に再生できないことを示すフラグデータである。

【0230】続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたピクチャ層のピクチャヘッダを表わすデータエレメントが、履歴ストリームとして記述されている。

【0231】このピクチャヘッダに関するデータエレメントは、picture_start_code、temporal_reference、picture_coding_type、vbv_delay、full_pel_forward_vector、forward_f_code、full_pel_backward_vector、及び backward_f_codeから構成される。

【0232】具体的には、picture_start_codeは、ピクチャ層の開始同期コードを表すデータである。temporal_referenceは、ピクチャの表示順を示す番号でGOPの先頭でリセットされるデータである。picture_coding_typeは、ピクチャタイプを示すデータである。vbv_delayは、ランダムアクセス時の仮想バッファの初期状態を示すデータである。full_pel_forward_vectorは、順方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。forward_f_codeは、順方向動きベクトル探索範囲を表すデータである。full_pel_backward_vectorは、逆方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。backward_f_codeは、逆方向動きベクトル探索範囲を表すデータである。

【0233】続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたピクチャ層のピクチャコーディングエクステンションが、履歴ストリームとして記述されている。

【0234】このピクチャコーディングエクステンションに関するデータエレメントは、extension_start_code、extension_start_code_identifier、f_code[0][0]、f_code[0][1]、f_code[1][0]、f_code[1][1]、intra_dc_precision、picture_structure、top_field_first、frame_predictive_frame_dct、concealment_motion_vectors、q_scale_type、intra_vlc_format、alternate_scan、repeat_firt_field、chroma_420_type、progressive_frame、composite_display_flag、v_axis、field_sequence、sub_carrier、burst_amplitude、及びsub_carrier_phaseから構成される。

【0235】extension_start_codeは、ピクチャ層のエクステンションデータのスタートを示す開始コードであ

る。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。f_code[0][0]は、フォワード方向の水平動きベクトル探索範囲を表すデータである。f_code[0][1]は、フォワード方向の垂直動きベクトル探索範囲を表すデータである。f_code[1][0]は、バックワード方向の水平動きベクトル探索範囲を表すデータである。f_code[1][1]は、バックワード方向の垂直動きベクトル探索範囲を表すデータである。

【0236】intra_dc_precisionは、DC係数の精度を表すデータである。picture_structureは、フレームストラクチャかフィールドストラクチャかを示すデータである。フィールドストラクチャの場合は、上位フィールドか下位フィールドかもあわせて示すデータである。top_field_firstは、フレームストラクチャの場合、最初のフィールドが上位か下位かを示すデータである。frame_predictive_frame_dctは、フレーム・ストラクチャの場合、フレーム・モードDCTの予測がフレーム・モードだけであることを示すデータである。concealment_motion_vectorsは、イントラマクロブロックに伝送エラーを隠蔽するための動きベクトルがついていることを示すデータである。

【0237】q_scale_typeは、線形量子化スケールを利用するか、非線形量子化スケールを利用するかを示すデータである。intra_vlc_formatは、イントラマクロブロックに、別の2次元VLCを使うかどうかを示すデータである。alternate_scanは、ジグザグスキャンを使うか、オルタネート・スキャンを使うかの選択を表すデータである。repeat_firt_fieldは、2:3ブルダウの際に使われるデータである。chroma_420_typeは、信号フォーマットが4:2:0の場合、次のprogressive_frameと同じ値、そうでない場合は0を表すデータである。progressive_frameは、このピクチャが、順次走査できているかどうかを示すデータである。composite_display_flagは、ソース信号がコンポジット信号であったかどうかを示すデータである。

【0238】v_axisは、ソース信号が、PALの場合に使われるデータである。field_sequenceは、ソース信号が、PALの場合に使われるデータである。sub_carrierは、ソース信号が、PALの場合に使われるデータである。burst_amplitudeは、ソース信号が、PALの場合に使われるデータである。sub_carrier_phaseは、ソース信号が、PALの場合に使われるデータである。

【0239】続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用された量子化マトリックスエクステンションが、履歴ストリームとして記述されている。

【0240】この量子化マトリックスエクステンションに関するデータエレメントは、extension_start_code、extension_start_code_identifier、quant_matrix_exte

nsion_present_flag、load_intra_quantizer_matrix、intra_quantizer_matrix[64]、load_non_intra_quantizer_matrix、non_intra_quantizer_matrix[64]、load_chroma_intra_quantizer_matrix、chroma_intra_quantizer_matrix[64]、load_chroma_non_intra_quantizer_matrix、及びchroma_non_intra_quantizer_matrix[64] から構成される。

【0241】extension_start_codeは、この量子化マトリックスエクステンションのスタートを示す開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。quant_matrix_extension_present_flagは、この量子化マトリックスエクステンション内のデータエレメントが有効か無効かを示すためのデータである。load_intra_quantizer_matrixは、イントラマクロブロック用の量子化マトリックスデータの存在を示すデータである。intra_quantizer_matrixは、イントラマクロブロック用の量子化マトリックスの値を示すデータである。

【0242】load_non_intra_quantizer_matrixは、非イントラマクロブロック用の量子化マトリックスデータの存在を示すデータである。non_intra_quantizer_matrixは、非イントラマクロブロック用の量子化マトリックスの値を表すデータである。load_chroma_intra_quantizer_matrixは、色差イントラマクロブロック用の量子化マトリックス・データの存在を示すデータである。chroma_intra_quantizer_matrixは、色差イントラマクロブロック用の量子化マトリックスの値を示すデータである。load_chroma_non_intra_quantizer_matrixは、色差非イントラマクロブロック用の量子化マトリックス・データの存在を示すデータである。chroma_non_intra_quantizer_matrixは、色差非イントラマクロブロック用の量子化マトリックスの値を示すデータである。

【0243】続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたコピーライトエクステンションが、履歴ストリームとして記述されている。

【0244】このコピーライトエクステンションに関するデータエレメントは、extension_start_code、extension_start_code_identifier、copyright_extension_present_flag、copyright_flag、copyright_identifier、original_or_copy、copyright_number_1、copyright_number_2、及び copyright_number_3から構成される。

【0245】extension_start_codeは、コピーライトエクステンションのスタートを示す開始コードである。extension_start_code_identifierのどのエクステンションデータが送られるかを示すコードである。copyright_extension_present_flagは、このコピーライトエクステンション内のデータエレメントが有効か無効かを示すためのデータである。copyright_flagは、次のコピーライ

トエクステンション又はシーケンスエンドまで、符号化されたビデオデータに対してコピー権が与えられているか否かを示す。

【0246】copyright_identifierは、ISO/IEC JTC/SC 29によって指定されたコピー権の登録機関を識別するためのデータである。original_or_copyは、ビットストリーム中のデータが、オリジナルデータであるかコピーデータであることを示すデータである。copyright_number_1は、コピーライトナンバーのビット44から63を表わすデータである。copyright_number_2は、コピーライトナンバーのビット22から43を表わすデータである。copyright_number_3は、コピーライトナンバーのビット0から21を表わすデータである。

【0247】続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたピクチャディスプレイエクステンション (picture_display_extension) が、履歴ストリームとして記述されている。

【0248】このピクチャディスプレイエクステンションを表わすデータエレメントは、extension_start_code、extension_start_code_identifier、picture_display_extension_present_flag、frame_center_horizontal_offset_1、frame_center_vertical_offset_1、frame_center_horizontal_offset_2、frame_center_vertical_offset_2、frame_center_horizontal_offset_3、及びframe_center_vertical_offset_3から構成される。

【0249】extension_start_codeは、ピクチャディスプレイエクステンションのスタートを示すための開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。picture_display_extension_present_flagは、ピクチャディスプレイエクステンション内のデータエレメントが有効か無効かを示すデータである。frame_center_horizontal_offsetは、表示エリアの水平方向のオフセットを示すデータであって、3つのオフセット値まで定義することができる。frame_center_vertical_offsetは、表示エリアを垂直方向のオフセットを示すデータであって、3つのオフセット値まで定義することができる。

【0250】最終段の符号化処理において生成されたビットストリームのピクチャ層のユーザエリアには、既に説明したピクチャディスプレイエクステンションを表わす履歴情報の次に、過去の符号化処理において使用されたユーザデータが、履歴ストリームとして記述されている。

【0251】このユーザデータの次には、過去の符号化処理において使用されたマクロブロック層に関する情報が、履歴ストリームとして記述されている。

【0252】このマクロブロック層に関する情報は、macroblock_address_h、macroblock_address_v、slice_header_present_flag、skipped_macroblock_flag等のマク

ロブロックの位置に関するデータエレメントと、macroblock_quant、macroblock_motion_forward、macroblock_motion_backward、macroblock_pattern、macroblock_intra、spatial_temporal_weight_code_flag、frame_motion_type、及びdct_type等のマクロブロックモードに関するデータエレメントと、quantiser_scale_code等の量子化ステップ制御に関するデータエレメントと、PMV[0][0]、PMV[0][0][1]、motion_vertical_field_select[0][0]、PMV[0][1][0]、PMV[0][1][1]、motion_vertical_field_select[0][1]、PMV[1][0][0]、PMV[1][0][1]、motion_vertical_field_select[1][0]、PMV[1][1][0]、PMV[1][1][1]、motion_vertical_field_select[1][1]等の動き補償に関するデータエレメントと、coded_block_pattern等のマクロブロックパターンに関するデータエレメントと、num_mv_bits、num_coef_bits、及びnum_other_bits等の発生符号量に関するデータエレメントから構成されている。

【0253】以下にマクロブロック層に関するデータエレメントについて詳細に説明する。

【0254】macroblock_address_hは、現在のマクロブロックの水平方向の絶対位置を定義するためのデータである。macroblock_address_vは、現在のマクロブロックの垂直方向の絶対位置を定義するためのデータである。slice_header_present_flagは、このマクロブロックがスライス層の先頭であり、スライスヘッダを伴うか否かを示すデータである。skipped_macroblock_flagは、復号化処理においてこのマクロブロックをスキップするか否かを示すデータである。

【0255】macroblock_quantは、後述する図70乃至図72に示されたマクロブロックタイプ(macroblock_type)から導かれるデータであって、quantiser_scale_codeがビットストリーム中に現れるか否かを示すデータである。macroblock_motion_forwardは、図70乃至図72に示されたマクロブロックタイプから導かれるデータであって、復号化処理で使用されるデータである。macroblock_motion_backwardは、図70乃至図72に示されたマクロブロックタイプから導かれるデータであって、復号化処理で使用されるデータである。macroblock_patternは、図70乃至図72に示されたマクロブロックタイプから導かれるデータであって、coded_block_patternがビットストリーム中に現れるか否かを示すデータである。

【0256】macroblock_intraは、図70乃至図72に示されたマクロブロックタイプから導かれるデータであって、復号化処理で使用されるデータである。spatial_temporal_weight_code_flagは、図70乃至図72に示されたマクロブロックタイプから導かれるデータであって、時間スケラビリティで下位レイヤ画像のアップサンプリング方法を示すspatial_temporal_weight_codeは、ビットストリーム中に存在するか否かを示すデータ

である。

【0257】frame_motion_typeは、フレームのマクロブロックの予測タイプを示す2ビットのコードである。予測ベクトルが2個でフィールドベースの予測タイプであれば「00」であって、予測ベクトルが1個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが1個でフレームベースの予測タイプであれば「10」であって、予測ベクトルが1個でディアルプライムの予測タイプであれば「11」である。field_motion_typeは、フィールドのマクロブロックの動き予測を示す2ビットのコードである。予測ベクトルが1個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが2個で18×8マクロブロックベースの予測タイプであれば「10」であって、予測ベクトルが1個でディアルプライムの予測タイプであれば「11」である。dct_typeは、DCTがフレームDCTモードか、フィールドDCTモードかを示すデータである。quantiser_scale_codeはマクロブロックの量子化ステップサイズを示すデータである。

【0258】次に動きベクトルに関するデータエレメントについて説明する。動きベクトルは、復号時に必要な動きベクトルを減少させるために、先に符号化されたベクトルに関し差分として符号化される。動きベクトルの復号を行うために復号器は、4個の動きベクトル予測値(それぞれ水平及び垂直成分を伴う)を維持しなければいけない。この予測動きベクトルをPMV[r][s][v]と表わすことにしている。[r]は、マクロブロックにおける動きベクトルが第1のベクトルであるのか、第2のベクトルであるのかを示すフラグであって、マクロブロックにおけるベクトルが第1のベクトルである場合には「0」となって、マクロブロックにおけるベクトルが第2のベクトルである場合には「1」となる。[s]は、マクロブロックにおける動きベクトルの方向が、前方向であるのか後方向であるのかを示すフラグであって、前方向動きベクトルの場合には「0」となって、後方向動きベクトルの場合には「1」となる。[v]は、マクロブロックにおけるベクトルの成分が、水平方向であるのか垂直方向であるのかを示すフラグであって、水平方向成分の場合には「0」となって、垂直方向成分の場合には「1」となる。

【0259】従って、PMV[0][0][0]は、第1のベクトルの前方向の動きベクトルの水平方向成分のデータを表わし、PMV[0][0][1]は、第1のベクトルの前方向の動きベクトルの垂直方向成分のデータを表わし、PMV[0][1][0]は、第1のベクトルの後方向の動きベクトルの水平方向成分のデータを表わし、PMV[0][1][1]は、第1のベクトルの後方向の動きベクトルの垂直方向成分のデータを表わし、PMV[1][0][0]は、第2のベクトルの前方向の動きベクトルの水平方向成分のデータを表わし、PMV[1][0][1]は、第2のベクトルの前方向の動きベクトルの垂

直方向成分のデータを表わし、PMV[1][1][0]は、第2のベクトルの後方向の動きベクトルの水平方向成分のデータを表わし、PMV[1][1][1]は、第2のベクトルの後方向の動きベクトルの垂直方向成分のデータを表わしている。

【0260】motion_vertical_field_select[r][s]は、予測の形式にいずれの参照フィールドを使用するかを示すデータである。このmotion_vertical_field_select[r][s]が「0」の場合には、トップ参照フィールドを使用し、「1」の場合には、ボトム参照フィールドを使用することを示している。

【0261】よって、motion_vertical_field_select[0][0]は、第1のベクトルの前方向の動きベクトルを生成する際の参照フィールドを示し、motion_vertical_field_select[0][1]は、第1のベクトルの後方向の動きベクトルを生成する際の参照フィールドを示し、motion_vertical_field_select[1][0]は、第2のベクトルの前方向の動きベクトルを生成する際の参照フィールドを示し、motion_vertical_field_select[1][1]は、第2ベクトルの後方向の動きベクトルを生成する際の参照フィールドを示している。

【0262】coded_block_patternは、DCT係数を格納する複数のDCTブロックのうち、どのDCTブロックに、有意係数（非0係数）があるかを示す可変長のデータである。num_mv_bitsは、マクロブロック中の動きベクトルの符号量を示すデータである。num_coef_bitsは、マクロブロック中のDCT係数の符号量を示すデータである。num_other_bitsは、マクロブロックの符号量で、動きベクトル及びDCT係数以外の符号量を示すデータである。

【0263】次に、可変長の履歴ストリームから各データエレメントをデコードするためのシンタックスについて、図50乃至図69を参照して説明する。

【0264】この可変長の履歴ストリームは、next_start_code()関数、sequence_header()関数、sequence_extension()関数、extension_and_user_data(0)関数、group_of_picture_header()関数、extension_and_user_data(1)関数、picture_header()関数、picture_coding_extension()関数、re_coding_stream_info()関数、extension_and_user_data(2)関数、及びpicture_data()関数によって定義されたデータエレメントによって構成される。

【0265】next_start_code()関数は、ビットストリーム中に存在するスタートコードを探すための関数であるので、履歴ストリームの最も先頭には、図51に示すような、過去の符号化処理において使用されたデータエレメントであってsequence_header()関数によって定義されたデータエレメントが記述されている。

【0266】sequence_header()関数によって定義されたデータエレメントは、sequence_header_code、sequence_header_present_flag、horizontal_size_value、vertical_size_value、aspect_ratio_information、frame_

rate_code、bit_rate_value、marker_bit、VBV_buffer_size_value、constrained_parameter_flag、load_intra_quantizer_matrix、intra_quantizer_matrix、load_non_intra_quantizer_matrix、及びnon_intra_quantizer_matrix等である。

【0267】sequence_header_codeは、シーケンス層のスタート同期コードを表すデータである。sequence_header_present_flagは、sequence_header内のデータが有効か無効かを示すデータである。horizontal_size_valueは、画像の水平方向の画素数の下位12ビットから成るデータである。vertical_size_valueは、画像の縦のライン数の下位12ビットからなるデータである。aspect_ratio_informationは、画素のアスペクト比（縦横比）または表示画面アスペクト比を表すデータである。frame_rate_codeは、画像の表示周期を表すデータである。bit_rate_valueは、発生ビット量に対する制限のためのビット・レートの下位18ビット（400bsp単位で切り上げる）データである。

【0268】marker_bitは、スタートコードエミュレーションを防止するために挿入されるビットデータである。VBV_buffer_size_valueは、発生符号量制御用の仮想バッファ（ビデオバッファペリファイヤー）の大きさを決める値の下位10ビットデータである。constrained_parameter_flagは、各パラメータが制限以内であることを示すデータである。load_intra_quantizer_matrixは、イントラMB用量子化マトリックス・データの存在を示すデータである。intra_quantizer_matrixは、イントラMB用量子化マトリックスの値を示すデータである。load_non_intra_quantizer_matrixは、非イントラMB用量子化マトリックス・データの存在を示すデータである。non_intra_quantizer_matrixは、非イントラMB用量子化マトリックスの値を表すデータである。

【0269】sequence_header()関数によって定義されたデータエレメントの次には、図52で示すような、sequence_extension()関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。

【0270】sequence_extension()関数によって定義されたデータエレメントとは、extension_start_code、extension_start_code_identifier、sequence_extension_present_flag、profile_and_level_indication、progressive_sequence、chroma_format、horizontal_size_extension、vertical_size_extension、bit_rate_extension、vbv_buffer_size_extension、low_delay、frame_rate_extension_n、及びframe_rate_extension_d等のデータエレメントである。

【0271】extension_start_codeは、エクステンションデータのスタート同期コードを表すデータである。extension_start_code_identifierは、どの拡張データが送られるかを示すデータである。sequence_extension_present_flagは、シーケンスエクステンション内のデー

タが有効であるか無効であるかを示すデータである。profile_and_level_indicationは、ビデオデータのプロファイルとレベルを指定するためのデータである。progressive_sequenceは、ビデオデータが順次走査であることを示すデータである。chroma_formatは、ビデオデータの色差フォーマットを指定するためのデータである。horizontal_size_extensionは、シーケンスヘッダのhorizontal_size_valueに加える上位2ビットのデータである。vertical_size_extensionは、シーケンスヘッダのvertical_size_valueに加える上位2ビットのデータである。bit_rate_extensionは、シーケンスヘッダのbit_rate_valueに加える上位12ビットのデータである。vbv_buffer_size_extensionは、シーケンスヘッダのvbv_buffer_size_valueに加える上位8ビットのデータである。

【0272】low_delayは、Bピクチャを含まないことを示すデータである。frame_rate_extension_nは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを取得するためのデータである。frame_rate_extension_dは、シーケンスヘッダのframe_rate_codeと組み合わせてフレームレートを取得するためのデータである。

【0273】sequence_extension()関数によって定義されたデータエレメントの次には、図53に示すようなextension_and_user_data(0)関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。extension_and_user_data(i)関数は、「i」が1以外のときは、extension_data()関数によって定義されるデータエレメントは記述せずに、user_data()関数によって定義されるデータエレメントのみを履歴ストリームとして記述する。よって、extension_and_user_data(0)関数は、user_data()関数によって定義されるデータエレメントのみを履歴ストリームとして記述する。

【0274】user_data()関数は、図54に示されたようなシンタックスに基いて、ユーザデータを履歴ストリームとして記述する。

【0275】extension_and_user_data(0)関数によって定義されたデータエレメントの次には、図55に示すようなgroup_of_picture_header()関数によって定義されたデータエレメント、及びextension_and_user_data(1)関数によって定義されるデータエレメントが、履歴ストリームとして記述されている。但し、履歴ストリーム中に、GOP層のスタートコードを示すgroup_start_codeが記述されている場合にのみ、group_of_picture_header()関数によって定義されたデータエレメント、及びextension_and_user_data(1)関数によって定義されるデータエレメントが記述されている。

【0276】group_of_picture_header()関数によって定義されたデータエレメントは、group_start_code、group_of_picture_header_present_flag、time_code、closed_gop、及びbroken_linkから構成される。

【0277】group_start_codeは、GOP層の開始同期コ

ードを示すデータである。group_of_picture_header_present_flagは、group_of_picture_header内のデータエレメントが有効であるか無効であるかを示すデータである。time_codeは、GOPの先頭ピクチャのシーケンスの先頭からの時間を示すタイムコードである。closed_gopは、GOP内の画像が他のGOPから独立再生可能なことを示すフラグデータである。broken_linkは、編集などのためにGOP内の先頭のBピクチャが正確に再生できないことを示すフラグデータである。

10 【0278】extension_and_user_data(1)関数は、extension_and_user_data(0)関数と同じように、user_data()関数によって定義されるデータエレメントのみを履歴ストリームとして記述する。

【0279】もし、履歴ストリーム中に、GOP層のスタートコードを示すgroup_start_codeが存在しない場合には、これらのgroup_of_picture_header()関数及びextension_and_user_data(1)関数によって定義されるデータエレメントは、履歴ストリーム中には記述されていない。その場合には、extension_and_user_data(0)関数
20 によって定義されたデータエレメントの次に、picture_headr()関数によって定義されたデータエレメントが履歴ストリームとして記述されている。

【0280】picture_headr()関数によって定義されたデータエレメントは、図56に示すように、picture_start_code、temporal_reference、picture_coding_type、vbv_delay、full_pel_forward_vector、forward_f_code、full_pel_backward_vector、backward_f_code、extra_bit_picture、及びextra_information_pictureである。

30 【0281】具体的には、picture_start_codeは、ピクチャ層の開始同期コードを表すデータである。temporal_referenceは、ピクチャの表示順を示す番号でGOPの先頭でリセットされるデータである。picture_coding_typeは、ピクチャタイプを示すデータである。vbv_delayは、ランダムアクセス時の仮想バッファの初期状態を示すデータである。full_pel_forward_vectorは、順方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。forward_f_codeは、順方向動きベクトル探索範囲を表すデータである。full_pel_backward_vector
40 は、逆方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。backward_f_codeは、逆方向動きベクトル探索範囲を表すデータである。extra_bit_pictureは、後続する追加情報の存在を示すフラグである。このextra_bit_pictureが「1」の場合には、次にextra_information_pictureが存在し、extra_bit_pictureが「0」の場合には、これに続くデータが無いことを示している。extra_information_pictureは、規格において予約された情報である。

50 【0282】picture_headr()関数によって定義されたデータエレメントの次には、図57に示すようなpictur

e_coding_extension()関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。

【0283】このpicture_coding_extension()関数によって定義されたデータエレメントとは、extension_start_code、extension_start_code_identifier、f_code[0][0]、f_code[0][1]、f_code[1][0]、f_code[1][1]、intra_dc_precision、picture_structure、top_field_first、frame_predictive_frame_dct、concealment_motion_vectors、q_scale_type、intra_vlc_format、alternate_scan、repeat_firt_field、chroma_420_type、progressive_frame、composite_display_flag、v_axis、field_sequence、sub_carrier、burst_amplitude、及びsub_carrier_phaseから構成される。

【0284】extension_start_codeは、ピクチャ層のエクステンションデータのスタートを示す開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。f_code[0][0]は、フォワード方向の水平動きベクトル探索範囲を表すデータである。f_code[0][1]は、フォワード方向の垂直動きベクトル探索範囲を表すデータである。f_code[1][0]は、バックワード方向の水平動きベクトル探索範囲を表すデータである。f_code[1][1]は、バックワード方向の垂直動きベクトル探索範囲を表すデータである。intra_dc_precisionは、DC係数の精度を表すデータである。

【0285】picture_structureは、フレームストラクチャかフィールドストラクチャかを示すデータである。フィールドストラクチャの場合は、上位フィールドか下位フィールドかもあわせて示すデータである。top_field_firstは、フレームストラクチャの場合、最初のフィールドが上位か下位かを示すデータである。frame_predictive_frame_dctは、フレーム・ストラクチャの場合、フレーム・モードDCTの予測がフレーム・モードだけであることを示すデータである。concealment_motion_vectorsは、イントラマクロブロックに伝送エラーを隠蔽するための動きベクトルがついていることを示すデータである。q_scale_typeは、線形量子化スケールを利用するか、非線形量子化スケールを利用するかを示すデータである。intra_vlc_formatは、イントラマクロブロックに、別の2次元VLCを使うかどうかを示すデータである。

【0286】alternate_scanは、ジグザグスキャンを使うか、オルタネート・スキャンを使うかの選択を表すデータである。repeat_firt_fieldは、2:3ブルダウンの際に使われるデータである。chroma_420_typeは、信号フォーマットが4:2:0の場合、次のprogressive_frameと同じ値、そうでない場合は0を表すデータである。progressive_frameは、このピクチャが、順次走査できているかどうかを示すデータである。composite_display_flagは、ソース信号がコンポジット信号であった

かどうかを示すデータである。v_axisは、ソース信号が、PALの場合に使われるデータである。field_sequenceは、ソース信号が、PALの場合に使われるデータである。sub_carrierは、ソース信号が、PALの場合に使われるデータである。burst_amplitudeは、ソース信号が、PALの場合に使われるデータである。sub_carrier_phaseは、ソース信号が、PALの場合に使われるデータである。

【0287】picture_coding_extension()関数によって定義されたデータエレメントの次には、re_coding_stream_info()関数によって定義されたデータエレメントが履歴ストリームとして記述されている。このre_coding_stream_info()関数は、主に履歴情報の組み合わせを記述する場合に用いられるものであり、その詳細については、図76を参照して後述する。

【0288】re_coding_stream_info()関数によって定義されたデータエレメントの次には、extensions_and_user_data(2)によって定義されたデータエレメントが、履歴ストリームとして記述されている。このextension_and_user_data(2)関数は、図53に示したように、ビットストリーム中にエクステンションスタートコード(extension_start_code)が存在する場合には、extension_data()関数によって定義されるデータエレメントが記述されている。このデータエレメントの次には、ビットストリーム中にユーザデータスタートコード(user_data_start_code)が存在する場合には、user_data()関数によって定義されるデータエレメントが記述されている。但し、ビットストリーム中にエクステンションスタートコード及びユーザデータスタートコードが存在しない場合には extension_data()関数 及びuser_data()関数によって定義されるデータエレメントはビットストリーム中には記述されていない。

【0289】extension_data()関数は、図58に示すように、extension_start_codeを示すデータエレメントと、sequence_display_extension()関数、quant_matrix_extension()関数、copyright_extension()関数、及びpicture_display_extension()関数によって定義されるデータエレメントとを、ビットストリーム中に履歴ストリームとして記述するための関数である。

【0290】sequence_display_extension()関数は、extension_data(i)関数が、i=0のときに存在する。iの値は、extension_data()が、group_of_pictures_header()に続くことはないので、1となることはない。

【0291】sequence_display_extension()関数は、図59に示すように、extension_start_code_identifier、video_format、colour_description、colour_primitives、transfer_characteristics、matrix_coefficients、display_horizontal_size、marker_bit、display_vertical_sizeのデータエレメントより構成されている。

【0292】extension_start_code_identifierは、拡

張 (extension) を識別するための 4 ビットの整数である。video_format は、符号化される前の画像の表示ビットの整数である。colour_description は、ビットストリーム中に、colour_primaries、transfer_characteristics、および matrix_coefficients が存在するか否かを表し、その値が 1 であるとき、これらが存在することを意味する。

【0293】 colour_primaries は、8 ビットの整数であり、情報元原色の色度座標を表す。transfer_characteristics は、8 ビットの整数であり、情報元画像の光電子変換特性を示す。matrix_coefficients は、8 ビットの整数であり、緑、青、および赤の原色から、輝度および色差信号を導くとき使用されるマトリクス係数を表す。これらの colour_primaries、transfer_characteristics、matrix_coefficients は、colour_description の値が 1 である場合のみ存在する。

【0294】 display_horizontal_size と display_vertical_size は、意図された表示の有効領域の水平方向の大きさと垂直方向の大きさを矩形で定義する。display_horizontal_size は、horizontal_size と同じ単位で定義され、display_vertical_size は、vertical_size と同じ単位で定義される。marker_bit は、スタートコードエミュレーションを防止するために挿入されるビットデータである。

【0295】 quant_matrix_extension() 関数によって定義されるデータエレメントは、図 60 に示すように、extension_start_code、extension_start_code_identifier、quant_matrix_extension_present_flag、load_intra_quantizer_matrix、intra_quantizer_matrix[64]、load_non_intra_quantizer_matrix、non_intra_quantizer_matrix[64]、load_chroma_intra_quantizer_matrix、chroma_intra_quantizer_matrix[64]、load_chroma_non_intra_quantizer_matrix、及び chroma_non_intra_quantizer_matrix[64] である。

【0296】 extension_start_code は、この量子化マトリクスエクステンションのスタートを示す開始コードである。extension_start_code_identifier は、どの拡張データが送られるかを示すコードである。quant_matrix_extension_present_flag は、この量子化マトリクスエクステンション内のデータエレメントが有効か無効かを示すためのデータである。load_intra_quantizer_matrix は、イントラマクロブロック用の量子化マトリクスデータの存在を示すデータである。intra_quantizer_matrix は、イントラマクロブロック用の量子化マトリクスの値を示すデータである。

【0297】 load_non_intra_quantizer_matrix は、非イントラマクロブロック用の量子化マトリクスデータの存在を示すデータである。non_intra_quantizer_matrix は、非イントラマクロブロック用の量子化マトリクスの値を表すデータである。load_chroma_intra_quantizer

zer_matrix は、色差イントラマクロブロック用の量子化マトリクス・データの存在を示すデータである。chroma_intra_quantizer_matrix は、色差イントラマクロブロック用の量子化マトリクスの値を示すデータである。load_chroma_non_intra_quantizer_matrix は、色差非イントラマクロブロック用の量子化マトリクス・データの存在を示すデータである。chroma_non_intra_quantizer_matrix は、色差非イントラマクロブロック用の量子化マトリクスの値を示すデータである。

10 【0298】 copyright_extension() 関数によって定義されるデータエレメントは、図 61 に示すように、extension_start_code、extension_start_code_identifier、copyright_extension_present_flag、copyright_flag、copyright_identifier、original_or_copy、copyright_number_1、copyright_number_2、及び copyright_number_3 から構成される。

20 【0299】 extension_start_code は、コピーライトエクステンションのスタートを示す開始コードである。extension_start_code_identifier どのエクステンションデータが送られるかを示すコードである。copyright_extension_present_flag は、このコピーライトエクステンション内のデータエレメントが有効か無効かを示すためのデータである。

【0300】 copyright_flag は、次のコピーライトエクステンション又はシーケンスエンドまで、符号化されたビデオデータに対してコピー権が与えられているか否かを示す。copyright_identifier は、ISO/IEC JTC/SC29 によって指定されたコピー権の登録機関を識別するためのデータである。original_or_copy は、ビットストリーム中のデータが、オリジナルデータであるかコピーデータであるかを示すデータである。copyright_number_1 は、コピーライトナンバーのビット 44 から 63 を表わすデータである。copyright_number_2 は、コピーライトナンバーのビット 22 から 43 を表わすデータである。copyright_number_3 は、コピーライトナンバーのビット 0 から 21 を表わすデータである。

40 【0301】 picture_display_extension() 関数によって定義されるデータエレメントは、図 62 に示すように、extension_start_code_identifier、frame_center_horizontal_offset、frame_center_vertical_offset 等である。

50 【0302】 extension_start_code_identifier は、どの拡張データが送られるかを示すコードである。frame_center_horizontal_offset は、表示エリアの水平方向のオフセットを示すデータであって、number_of_frame_center_offsets によって定義される数のオフセット値を定義することができる。frame_center_vertical_offset は、表示エリアを垂直方向のオフセットを示すデータであって、number_of_frame_center_offsets によって定義される数のオフセット値を定義することができる。

【0303】再び図50に戻って、extension_and_user_data(2)関数によって定義されるデータエレメントの次には、picture_data()関数によって定義されるデータエレメントが、履歴ストリームとして記述されている。但し、このpicture_data()関数は、red_bw_flagが1ではないか、または、red_bw_indicatorが2以下である場合に存在する。このred_bw_flagとred_bw_indicatorは、re_coding_stream_info()関数に記述されており、これらについては、図76と図77を参照して後述する。

【0304】picture_data()関数によって定義されるデータエレメントは、図63に示すように、slice()関数によって定義されるデータエレメントである。このslice()関数によって定義されるデータエレメントはビットストリーム中に少なくとも1個記述されている。

【0305】slice()関数は、図64に示されるように、slice_start_code、slice_quantiser_scale_code、intra_slice_flag、intra_slice、reserved_bits、extra_bit_slice、extra_information_slice、及びextra_bit_slice等のデータエレメントと、macroblock()関数によって定義されるデータエレメントを、履歴ストリームとして記述するための関数である。

【0306】slice_start_codeは、slice()関数によって定義されるデータエレメントのスタートを示すスタートコードである。slice_quantiser_scale_codeは、このスライス層に存在するマクロブロックに対して設定された量子化ステップサイズを示すデータである。しかし、各マクロブロック毎に、quantiser_scale_codeが設定されている場合には、各マクロブロックに対して設定されたmacroblock_quantiser_scale_codeのデータが優先して使用される。

【0307】intra_slice_flagは、ビットストリーム中にintra_slice及びreserved_bitsが存在するか否かを示すフラグである。intra_sliceは、スライス層中にノンイントラマクロブロックが存在するか否かを示すデータである。スライス層におけるマクロブロックのいずれかがノンイントラマクロブロックである場合には、intra_sliceは「0」となり、スライス層におけるマクロブロックの全てがノンイントラマクロブロックである場合には、intra_sliceは「1」となる。reserved_bitsは、7ビットのデータであって「0」の値を取る。extra_bit_sliceは、履歴ストリームとして追加の情報が存在することを示すフラグであって、次にextra_information_sliceが存在する場合には「1」に設定される。追加の情報が存在しない場合には「0」に設定される。

【0308】これらのデータエレメントの次には、macroblock()関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。

【0309】macroblock()関数は、図65に示すように、macroblock_escape、macroblock_address_increment、及びmacroblock_quantiser_scale_code、及びmarker

_bit等のデータエレメントと、macroblock_modes()関数、motion_vectors(s)関数、及びcode_block_pattern()関数によって定義されたデータエレメントを記述するための関数である。

【0310】macroblock_escapeは、参照マクロブロックと前のマクロブロックとの水平方向の差が34以上であるか否かを示す固定ビット列である。参照マクロブロックと前のマクロブロックとの水平方向の差が34以上の場合には、macroblock_address_incrementの値に33をプラスする。macroblock_address_incrementは、参照マクロブロックと前のマクロブロックとの水平方向の差を示すデータである。もし、このmacroblock_address_incrementの前にmacroblock_escapeが1つ存在するのであれば、このmacroblock_address_incrementの値に33をプラスした値が、実際の参照マクロブロックと前のマクロブロックとの水平方向の差分を示すデータとなる。

【0311】macroblock_quantiser_scale_codeは、各マクロブロック毎に設定された量子化ステップサイズであり、macroblock_quantが「1」のときだけ存在する。各スライス層には、スライス層の量子化ステップサイズを示すslice_quantiser_scale_codeが設定されているが、参照マクロブロックに対してmacroblock_quantiser_scale_codeが設定されている場合には、この量子化ステップサイズを選択する。

【0312】macroblock_address_incrementの次には、macroblock_modes()関数によって定義されるデータエレメントが記述されている。macroblock_modes()関数は、図66に示すように、macroblock_type、frame_motion_type、field_motion_type、dct_type等のデータエレメントを、履歴ストリームとして記述するための関数である。

【0313】macroblock_typeは、マクロブロックの符号化タイプを示すデータである。具体的には、図67乃至図69に示されるように、macroblock_typeは、macroblock_quant、dct_type_flag、macroblock_motion_forward、及びmacroblock_motion_backwardなどのフラグから生成された可変長データである。macroblock_quantは、マクロブロックに対して量子化ステップサイズを設定するためのmacroblock_quantiser_scale_codeが設定されているか否かを示すフラグであって、ビットストリーム中にmacroblock_quantiser_scale_codeが存在する場合には、macroblock_quantは「1」の値を取る。

【0314】dct_type_flagは、参照マクロブロックがフレームDCT又はフィールドDCTで符号化されているかを示すdct_typeが存在するか否かを示すためのフラグ（言い換えるとDCTされているか否かを示すフラグ）であって、ビットストリーム中にdct_typeが存在する場合には、このdct_type_flagは「1」の値を取る。macroblock_motion_forwardは、参照マクロブロックが前方予測されているか否かを示すフラグであって、前方予測され

ている場合には「1」の値を取る。macroblock_motion_backwardは、参照マクロブロックが後方予測されているか否かを示すフラグであって、後方予測されている場合には「1」の値を取る。

【0315】もし、red_bw_flagが「1」ではないか、または、red_bw_flagが1であり、且つ、red_bw_indicatorが1以下である場合（後述する図77の組み合わせが、組み合わせ1、2または3の場合）であり、macroblock_motion_forward又はmacroblock_motion_backwardが「1」であり、ピクチャ構造がフレームであり、さらにframe_period_frame_dctが「0」である場合には、macroblock_typeを表わすデータエレメントの次にframe_motion_typeを表わすデータエレメントが記述されている。尚、このframe_period_frame_dctは、frame_motion_typeがビットストリーム中に存在するか否かを示すフラグである。

【0316】frame_motion_typeは、フレームのマクロブロックの予測タイプを示す2ビットのコードである。予測ベクトルが2個でフィールドベースの予測タイプであれば「00」であって、予測ベクトルが1個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが1個でフレームベースの予測タイプであれば「10」であって、予測ベクトルが1個でディアルプライムの予測タイプであれば「11」である。

【0317】frame_motion_typeを記述する条件が満足されない場合には、macroblock_typeを表わすデータエレメントの次にfield_motion_typeを表わすデータエレメントが記述されている。

【0318】field_motion_typeは、フィールドのマクロブロックの動き予測を示す2ビットのコードである。予測ベクトルが1個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが2個で18×8マクロブロックベースの予測タイプであれば「10」であって、予測ベクトルが1個でディアルプライムの予測タイプであれば「11」である。

【0319】もし、ピクチャ構造がフレームで、frame_period_frame_dctがframe_motion_typeがビットストリーム中に存在することを示し、且つ、frame_period_frame_dctがdct_typeがビットストリーム中に存在することを示し、さらに、red_bw_flagが「1」ではないか、または、red_bw_flagが「1」で、red_bw_indicatorが1以下である場合には、macroblock_typeを表わすデータエレメントの次にdct_typeを表わすデータエレメントが記述されている。尚、dct_typeは、DCTがフレームDCTモードか、フィールドDCTモードかを示すデータである。

【0320】再び図65に戻って、もし、red_bw_flagが「1」ではないか、または、red_bw_flagが「1」で、red_bw_indicatorが1以下であり、さらに、参照マクロブロックが前方予測マクロブロックであるか又は参照マクロブロックがイントラマクロブロックであって且

つコンシール処理のマクロブロックのいずれかの場合には、motion_vectors(0)関数によって定義されるデータエレメントが記述される。また、参照マクロブロックが後方予測マクロブロックである場合には、motion_vectors(1)関数によって定義されるデータエレメントが記述される。尚、motion_vectors(0)関数は、第1番めの動きベクトルに関するデータエレメントを記述するための関数であって、motion_vectors(1)関数は、第2番めの動きベクトルに関するデータエレメントを記述するための関数である。

【0321】motion_vectors(s)関数は、図70に示されるように、動きベクトルに関するデータエレメントを記述するための関数である。

【0322】もし、動きベクトルが1個でディアルプライム予測モードを使用していない場合には、motion_vertical_field_select[0][s]とmotion_vector(0, s)によって定義されるデータエレメントが記述される。

【0323】このmotion_vertical_field_select[r][s]は、第1番目の動きベクトル（前方又は後方のどちらのベクトルであっても良い）が、ボトムフィールドを参照して作られたベクトルであるかトップフィールドを参照して作られたベクトルであることを示すフラグである。この指標“r”は、第1番めのベクトル又は第2番めのベクトルのいずれのベクトルであることを示す指標であって、“s”は、予測方向が前方又は後方予測のいずれであることを示す指標である。

【0324】motion_vector(r, s)関数は、図71に示されるように、motion_code[r][s][t]に関するデータ列と、motion_residual[r][s][t]に関するデータ列と、dmvector[t]を表わすデータとを記述するための関数である。

【0325】motion_code[r][s][t]は、動きベクトルの大きさを-16～+16の範囲で表わす可変長のデータである。motion_residual[r][s][t]は、動きベクトルの残差を表わす可変長のデータである。よって、このmotion_code[r][s][t]とmotion_residual[r][s][t]との値によって詳細な動きベクトルを記述することができる。dmvector[t]は、ディアルプライム予測モードのときに、一方のフィールド（例えばボトムフィールドに対してトップフィールドを一方のフィールドとする）における動きベクトルを生成するために、時間距離に応じて既存の動きベクトルがスケールされると共に、トップフィールドとボトムフィールドとのライン間の垂直方向のずれを反映させるために垂直方向に対して補正を行うデータである。この指標“r”は、第1番めのベクトル又は第2番めのベクトルのいずれのベクトルであることを示す指標であって、“s”は、予測方向が前方又は後方予測のいずれであることを示す指標である。“s”は、動きベクトルが垂直方向の成分であるか水平方向の成分であるかを示すデータである。

【0326】図71に示されmotion_vector(r,s)関数によって、まず、水平方向のmotion_coder[r][s][0]を表わすデータ列が、履歴ストリームとして記述される。motion_residual[0][s][t]及びmotion_residual[1][s][t]の双方のビット数は、f_code[s][t]で示されるので、f_code[s][t]が1でない場合には、motion_residual[r][s][t]がビットストリーム中に存在することを示すことになる。水平方向成分のmotion_residual[r][s][0]が「1」でなくて、水平方向成分のmotion_coder[r][s][0]が「0」でないということは、ビットストリーム中にmotion_residual[r][s][0]を表わすデータエレメントが存在し、動きベクトルの水平方向成分が存在するということを示している。その場合には、水平方向成分のmotion_residual[r][s][0]を表わすデータエレメントが記述されている。

【0327】続いて、垂直方向のmotion_coder[r][s][1]を表わすデータ列が、履歴ストリームとして記述される。同じようにmotion_residual[0][s][t]及びmotion_residual[1][s][t]の双方のビット数は、f_code[s][t]で示されるので、f_code[s][t]が1でない場合には、motion_residual[r][s][t]がビットストリーム中に存在することを表わすことになる。motion_residual[r][s][1]が「1」でなくて、motion_coder[r][s][1]が「0」でないということは、ビットストリーム中にmotion_residual[r][s][1]を表わすデータエレメントが存在し、動きベクトルの垂直方向成分が存在するということを示している。その場合には、垂直方向成分のmotion_residual[r][s][1]を表わすデータエレメントが記述されている。

【0328】図65に戻って、coded_block_pattern()関数は、macroblock_patternが「1」であり、且つ、red_bw_flagが「1」ではないか、または、red_bw_flagが「1」で、red_bw_indicatorが0である場合には、coded_block_pattern()関数が、marker_bitの次に記述される。これらの条件が満足されない場合には、coded_block_pattern()関数は記述されない。

【0329】coded_block_pattern()関数は、図72に示すように、coded_block_pattern_420, coded_block_pattern_1, coded_block_pattern_2のデータエレメントより構成されている。coded_block_pattern_420は、変数cbpを導くために使用される可変長コードである。coded_block_pattern_1は、chroma_formatが4:2:2である場合に、2ビットの固定長コードを付加する場合のものであり、coded_block_pattern_2は、chroma_formatが4:4:4である場合に、6ビットの固定長コードを付加するものである。

【0330】なお、可変長フォーマットにおいては、伝送するビットレートを減少させるために、履歴情報を削減することができる。

【0331】すなわち、macroblock_typeとmotion_vect

ors()は転送するが、quantiser_scale_codeを転送しない場合には、slice_quantiser_scale_codeを”0000”とすることで、ビットレートを減少させることができる。

【0332】また、macroblock_typeのみ転送し、motion_vectors(), quantiser_scale_code, およびdct_typeを転送しない場合には、macroblock_typeとして、”not coded”を使用することで、ビットレートを減少することができる。

10 【0333】さらにまた、picture_coding_typeのみ転送し、slice()以下の情報は全て転送しない場合には、slice_start_codeを持たないpicture_data()を使用することで、ビットレートを減少させることができる。

【0334】以上においては、user_data内の23ビットの連続する”0”が出ないようにする場合に、22ビット毎に”1”を挿入するようにしたが、22ビット毎でなくてもよい。また、連続する”0”の個数を数えて”1”を挿入するのではなく、Byte_alignを調べて挿入するようにすることも可能である。

20 【0335】さらに、MPEGにおいては、23ビットの連続する”0”の発生を禁止しているが、実際には、バイトの先頭から23ビット連続する場合だけが問題とされ、バイトの先頭ではなく、途中から0が23ビット連続する場合は、問題とされない。従って、例えば24ビット毎に、LSB以外の位置に”1”を挿入するようにしてもよい。

30 【0336】また、以上においては、履歴情報を、video elementary streamに近い形式にしたが、packetized elementary streamやtransport streamに近い形式にしてもよい。また、Elementary Streamのuser_dataの場所を、picture_dataの前としたが、他の場所にすることもできる。

【0337】図18のトランスコード101においては、4世代分の符号化パラメータを履歴情報として後段に出力するようにしたが、実際には、履歴情報の全てが必要となるわけではなく、アプリケーション毎に必要な履歴情報は異なってくる。また、実際の伝送路あるいは記録媒体(伝送メディア)には、容量に制限があり、圧縮しているとはいえ、全ての履歴情報を伝送するようにすると、容量的に負担となり、結果的に画像ビットストリームのビットレートを抑圧してしまい、履歴情報伝送の有効性が損なわれることになる。

【0338】そこで、履歴情報として伝送する項目の組み合わせを記述する記述子を履歴情報に組み込んで後段に送信するようにし、全ての履歴情報を伝送するのではなく、様々なアプリケーションに対応した情報を伝送するようにすることができる。図73は、このような場合のトランスコード101の構成例を表している。

50 【0339】図73において、図18における場合と対応する部分には同一の符号を付してあり、その説明は適

宜省略する。図 73 の構成例においては、符号化パラメータ分離装置 105 と符号化装置 106 の間、及び履歴符号化装置 107 と符号化装置 106 の間に、符号化パラメータ選択回路 501 が挿入されている。

【0340】符号化パラメータ選択回路 501 は、符号化パラメータ分離装置 105 が出力するベースバンドビデオ信号から符号化パラメータを算出する符号化パラメータ算出部 512、符号化パラメータ分離装置 105 が出力する、このトランスコード 101 において、符号化するのに最適と判定された符号化パラメータ (いまの場合、2 世代前の符号化パラメータ) に関する情報から、符号化パラメータと記述子 (red_bw_flag, red_bw_indicator) を分離する組合せ記述子分離部 511、並びに符号化パラメータ算出部 512 が出力する符号化パラメータと、組合せ記述子分離部 511 が出力する符号化パラメータのうち、いずれか一方を、組合せ記述子分離部 511 で分離された記述子に対応して選択し、符号化装置 106 に出力するスイッチ 513 を有している。その他の構成は、図 18 における場合と同様である。

【0341】ここで、履歴情報として伝送する項目の組み合わせについて説明する。履歴情報は、分類すると、picture 単位の情報と、macroblock 単位の情報に分けることができる。slice 単位の情報は、それに含まれる macroblock の情報を収集することで得ることができ、GOP 単位の情報は、それに含まれる picture 単位の情報を収集することで得ることができる。

【0342】picture 単位の情報は、1 フレーム毎に 1 回伝送されるだけなので、情報伝送に占めるビットレートは、それほど大きくはない。これに対して、macroblock 単位の情報は、各 macroblock 毎に伝送されるため、例えば 1 フレームの走査線数が 525 本で、フィールドレートが 60 フィールド/秒のビデオシステムの場合、1 フレームの画素数を 720×480 とすると、macroblock 単位の情報は、1 フレームあたり 1350 (= 720/16×480/16) 回伝送することが必要となる。このため、履歴情報の相当の部分が macroblock 毎の情報で占められることになる。そこで、履歴情報としては、少なくとも picture 単位の情報は常に伝送するが、macroblock 単位の情報は、アプリケーションに応じて選択して伝送するようにすることで、伝送する情報量を抑制することができる。

【0343】履歴情報として転送される macroblock 単位の情報には、例えば num_coef_bits, num_mv_bits, num_other_bits, q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[], mv[], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb などがある。これらは、macroblock rate information の要素を用いて表現されたものである。

【0344】num_coef_bits は、macroblock の符号量の

うち、DCT 係数に要した符号量を表す。num_mv_bits は、macroblock の符号量のうち、動きベクトルに要した符号量を表す。num_other_bits は、macroblock の符号量のうち、num_coef_bits 及び num_mv_bits 以外の符号量を表す。

【0345】q_scale_code は、macroblock に適用された q_scale_code を表す。motion_type は、macroblock に適用された動きベクトルの type を表す。mv_vert_field_sel[][] は、macroblock に適用された動きベクトルの field select を表す。

【0346】mv[][][] は、macroblock に適用された動きベクトルを表す。mb_mfwd は、macroblock の予測モードが前方向予測であることを示すフラグである。mb_mbwd は、macroblock の予測モードが後方向予測であることを示すフラグである。mb_pattern は、macroblock の DCT 係数の非 0 のものの有無を示すフラグである。

【0347】coded_block_pattern は、macroblock の DCT 係数の非 0 のものの有無を DCT ブロック 毎に示すフラグである。mb_intra は、macroblock が intra_macro かそうでないかを示すフラグである。slice_start は、macroblock が slice の先頭であるか否かを示すフラグである。dct_type は、macroblock が field_dct か frame_dct かを示すフラグである。

【0348】mb_quant は、macroblock が quantiser_scale_code を伝送するか否かを示すフラグである。skipped_mb は、macroblock が skipped macroblock であるか否かを示すフラグである。

【0349】これらの項目は、常に全て必要であるわけではなく、アプリケーションに応じて必要となる項目が変化する。例えば、num_coef_bits や slice_start といった項目は、再エンコードした際のビットストリームをできる限り元の形に戻したいという transparent という要求を有するアプリケーションにおいて必要となる。換言すれば、ビットレートを変更するようなアプリケーションにおいては、これらの項目は必要ではない。また、非常に伝送路の制限が厳しい場合には、各ピクチャの符号化タイプが判るだけでもよいようなアプリケーションも存在する。このような状況から、履歴情報を伝送する項目の組み合わせの例として、例えば図 74 に示すような組み合わせが考えられる。

【0350】図 74 において、各組み合わせの中の項目に対応する値「2」は、その情報が存在し、利用可能であることを意味し、「0」は、その情報が存在しないことを意味する。「1」は、他の情報の存在を補助する目的のため、あるいは、構文上存在するが、元のビットストリーム情報とは関係がないなど、その情報自身には意味がないことを表している。例えば、slice_start は、履歴情報を伝送する際の slice の先頭の macroblock において、「1」になるが、本来のビットストリームに対して、slice が必ずしも同一位置関係にあるわけではない

場合には、履歴情報としては無意味になる。

【0351】図74の例においては、(num_coef_bits, num_mv_bits, num_other_bits), (q_scale_code, q_scale_type), (motion_type, mv_vert_field_sel[], mv[], mv[]), (mb_mfwd, mb_mbwd), (mb_pattern), (coded_block_pattern), (mb_intra), (slice_start), (dct_type), (mb_quant), (skipped_m_b)の各項目の有無により、組み合わせ1乃至組み合わせ5の5つの組み合わせが用意されている。

【0352】組み合わせ1は、完全にtransparentなビットストリームを再構成することを目的とした組み合わせである。この組み合わせによれば、発生符号量情報を用いることによる精度の高いトランスコーディングが実現できる。組み合わせ2も、完全にtransparentなビットストリームを再構成することを目的とした組み合わせである。組み合わせ3は、完全にtransparentなビットストリームを再構成することはできないが、視覚的にはほぼtransparentなビットストリームを再構成できるようにするための組み合わせである。組み合わせ4は、transparentという観点からは組み合わせ3よりも劣るが、視覚上問題がないビットストリームの再構成ができる組み合わせである。組み合わせ5は、transparentという観点からは組み合わせ4よりも劣るが、少ない履歴情報でビットストリームの完全ではない再構成ができる組み合わせである。

【0353】これらの組み合わせのうち、組み合わせの番号の数字が小さいものほど、機能的には上位であるが、履歴を転送するのに必要となる容量が多くなる。従って、想定するアプリケーションと履歴に使用できる容量を考慮することによって、伝送する組み合わせを決定

【0354】次に、図75のフローチャートを参照して、図73のトランスコーダ101の動作について説明する。ステップS41において、トランスコーダ101の復号装置102は、入力されたビットストリームを復号し、そのビットストリームを符号化する際に使用された符号化パラメータを抽出し、その符号化パラメータを符号化パラメータ多重装置103に出力するとともに、復号したビデオデータをやはり符号化パラメータ多重装置103に出力する。ステップS42において、復号装置102にはまた、入力されたビットストリームからuser_dataを抽出し、履歴復号装置104に出力する。履歴復号装置104は、ステップS43において、入力されたuser_dataから、組み合わせ情報(記述子)を用いて、履歴情報としての符号化パラメータを抽出し、符号化パラメータ多重装置103に出力する。

【0355】符号化パラメータ多重装置103は、ステップS44において、ステップS41で取り出された復号装置102から供給される符号化パラメータと、ステップS43で履歴復号装置104が出力した符号化パラ

メータとを、復号装置102から供給されるベースバンドのビデオデータに、図21または図34に示すようなフォーマットに従って多重化し、符号化パラメータ分離装置105に出力する。

【0356】符号化パラメータ分離装置105は、ステップS45において、符号化パラメータ多重装置103より供給されたベースバンドのビデオデータから符号化パラメータを抽出し、その中から今回の符号化に最も適している符号化パラメータを選択し、記述子とともに、組合せ記述子分離部511に出力する。また、符号化パラメータ分離装置105は、今回の符号化に最適と判定された符号化パラメータ以外の符号化パラメータ(例えば、最適な符号化パラメータが2世代前の符号化パラメータであると判定された場合には、それ以外の第1世代、第3世代、及び第4世代の符号化パラメータ)を履歴符号化装置107に出力する。履歴符号化装置107は、符号化パラメータ分離装置105より入力された符号化パラメータをステップS46において、user_dataに記述し、そのuser_data(converted_history_stream())を符号化装置106に出力する。

【0357】符号化パラメータ選択回路501の組合せ記述子分離部511は、符号化パラメータ分離装置105より供給されたデータから、符号化パラメータと記述子を分離し、符号化パラメータをスイッチ513の一方の接点に供給する。スイッチ513の他方の接点には、符号化パラメータ算出部512が、符号化パラメータ分離装置100が出力するベースバンドのビデオデータから、符号化パラメータを算出し、供給している。スイッチ513は、ステップS48において、組合せ記述子分離部511が出力した記述子に対応して、組合せ記述子分離部511が出力した符号化パラメータ、または符号化パラメータ算出部512が出力した符号化パラメータのいずれかを選択し、符号化装置106に出力する。すなわち、スイッチ513では、組合せ記述子分離部511から供給された符号化パラメータが有効である場合には、それが出力する符号化パラメータが選択されるが、それが出力する符号化パラメータが無効であると判定された場合には、符号化パラメータ算出部512がベースバンドビデオを処理することで算出した符号化パラメータが選択される。この選択は、伝送メディアの容量に対応して行われる。

【0358】符号化装置106は、ステップS49において、スイッチ513から供給された符号化パラメータに基づいて、符号化パラメータ分離装置105より供給されたベースバンドビデオ信号を符号化する。また、ステップS50において、符号化装置106は、符号化したビットストリームに、履歴符号化装置107より供給されたuser_dataを多重化し、出力する。

【0359】このようにして、各履歴によって得られる符号化パラメータの組み合わせが異なっているような場

合でも、支障なくトランスコーディングすることが可能となる。

【0360】このように、履歴情報は、図41に示したように、ビデオストリームのuser_data()関数の一種としてのhistory_stream() (より正確には、converted_history_stream()) で伝送される。そのhistory_stream()のシンタックスは、図50に示した通りである。履歴情報の項目の組み合わせを表す記述子 (red_bw_flag, red_bw_indicator) 、およびMPEGのストリームではサポートされていない項目 (num_other_bits, num_mv_bits, num_coef_bits) は、この図50の中のre_coding_stream_info()関数により伝送される。

【0361】re_coding_stream_info()関数は、図76に示すように、user_data_start_code, re_coding_stream_info_ID, red_bw_flag, red_bw_indicator, marker_bit, num_other_bits, num_mv_bits, num_coef_bitsなどのデータエレメントより構成される。

【0362】user_data_start_codeは、user_dataが開始することを表すスタートコードである。re_coding_stream_info_IDは、16ビットの整数であり、re_coding_stream_info()関数の識別のために用いられる。その値は、具体的には、"1001 00011110 1100" (0x91ec) とされる。

【0363】red_bw_flagは、1ビットのフラグであり、履歴情報が全ての項目を伝送する場合には0とされ、このフラグの値が1である場合、このフラグに続くred_bw_indicatorを調べることにより、図74に示した組み合わせのうち、どの組み合わせで項目が送られているのかを決定することができる。

【0364】red_bw_indicatorは、2ビットの整数であり、項目の組み合わせを図77に示すように記述する。

【0365】即ち、図74に示した5つの組み合わせのうち、組み合わせ1の場合、red_bw_flagは0とされ、組み合わせ2乃至組み合わせ5のとき、red_bw_flagは1とされる。これに対して、red_bw_indicatorは、組み合わせ2の場合0とされ、組み合わせ3の場合1とされ、組み合わせ4の場合2とされ、組み合わせ5の場合3とされる。

【0366】従って、red_bw_indicatorは、red_bw_flagが1の場合に (組み合わせ2乃至組み合わせ5の場合に) 規定される。

【0367】さらに、図76に示すように、red_bw_flagが0である場合 (組み合わせ1の場合)、マクロブロック毎に、marker_bit, num_other_bits, num_mv_bits, num_coef_bitsが記述される。これら4つのデータエレメントは、組み合わせ2乃至組み合わせ5の場合 (red_bw_flagが1の場合) 規定されない。

【0368】図63に示したように、picture_data()関数は、1個以上のslice()関数から構成される。しかしながら、組み合わせ5の場合、picture_data()関数を含

めて、それ以下のシンタックス要素は伝送されない (図74)。この場合、履歴情報は、picture_typeなどのpicture単位の情報の伝送を意図したものとなる。

【0369】組み合わせ1乃至組み合わせ4の場合、図64に示したslice()関数が存在する。しかしながら、このslice()関数によって決定されるsliceの位置情報と、元のビットストリームのsliceの位置情報は、履歴情報の項目の組み合わせに依存する。組み合わせ1または組み合わせ2の場合、履歴情報の元となったビットストリームのsliceの位置情報と、slice()関数によって決定されるsliceの位置情報とは、同一である必要がある。

【0370】図65に示すmacroblock()関数のシンタックス要素は、履歴情報の項目の組み合わせに依存する。macroblock_escape, macroblock_address_increment, macroblock_modes()関数は、常に存在する。しかしながら、macroblock_escapeとmacroblock_address_incrementの情報としての有効性は、組み合わせによって決定される。履歴情報の項目の組み合わせが、組み合わせ1または組み合わせ2の場合、元のビットストリームのskip ped_mb情報と同じものが伝送される必要がある。

【0371】組み合わせ4の場合、motion_vectors()関数は存在しない。組み合わせ1乃至組み合わせ3の場合、macroblock_modes()関数のmacroblock_typeによって、motion_vectors()関数の存在が決定される。組み合わせ3または組み合わせ4の場合には、coded_block_pattern()関数は存在しない。組み合わせ1と組み合わせ2の場合、macroblock_modes()関数のmacroblock_typeによって、coded_block_pattern()関数の存在が決定される。

【0372】図66に示したmacroblock_modes()関数のシンタックス要素は、履歴情報の項目の組み合わせに依存する。macroblock_typeは、常に存在する。組み合わせが組み合わせ4である場合、flame_motion_type, field_motion_type, dct_typeは存在しない。

【0373】macroblock_typeより得られるパラメータの情報としての有効性は、履歴情報の項目の組み合わせによって決定される。

【0374】履歴情報の項目の組み合わせが組み合わせ1または組み合わせ2である場合、macroblock_quantは、元のビットストリームと同じである必要がある。組み合わせ3または組み合わせ4の場合、macroblock_quantは、macroblock()関数内のquantiser_scale_codeの存在を表し、元のビットストリームと同じである必要はない。

【0375】組み合わせが組み合わせ1乃至組み合わせ3である場合、macroblock_motion_forwardとmacroblock_motion_backwardは、元のビットストリームと同一である必要がある。組み合わせが組み合わせ4または組み合わせ5である場合、その必要はない。

【0376】組み合わせが組み合わせ1または組み合わせ2である場合、macroblock_patternは、元のビットストリームと同一である必要がある。組み合わせ3の場合、macroblock_patternは、dct_typeの存在を示すのに用いられる。組み合わせが組み合わせ4である場合、組み合わせ1乃至組み合わせ3における場合のような関係は成立しない。

【0377】履歴情報の項目の組み合わせが組み合わせ1乃至組み合わせ3の場合、macroblock_intraは、元のビットストリームと同一である必要がある。組み合わせ4の場合には、その限りでない。

【0378】図50のhistory_stream()は、履歴情報を可変長とする場合のシンタックスであるが、図43乃至図44に示すように、固定長のシンタックスとする場合、固定長の履歴情報内に、伝送される項目中のどれが有効であるかを示す情報としての記述子(red_bw_flagとred_bw_indicator)をベースバンド画像に重畳し、伝送するようにする。その結果、この記述子を調べることにより、フィールドとして存在するが、その内容は無効であるといった判断をすることが可能となる。

【0379】このため、図47に示すように、re_coding_stream_informationとして、user_data_start_code, re_coding_stream_info_ID, red_bw_flag, red_bw_indicator, marker_bitが配置されている。それぞれの意味は、図76における場合と同様である。

【0380】このように履歴として伝送する符号化パラメータの要素をアプリケーションに応じた組み合わせで伝送するようにすることで、アプリケーションに応じた履歴を適当なデータ量で伝送するようにすることができる。

【0381】以上のように、履歴情報を可変長符号として伝送する場合、re_coding_stream_info()関数は、図76に示すように構成され、図50に示すように、history_stream()関数の一部として伝送される。これに対して、履歴情報を固定長符号として伝送する場合には、図47に示したように、history_stream()関数の一部として、re_coding_stream_information()が伝送される。図47の例では、re_coding_stream_informationとして、user_data_start_code, re_coding_stream_info_ID, red_bw_flag, red_bw_indicatorが伝送される。

【0382】また、図73の符号化パラメータ多重装置103が出力するベースバンドの信号中における履歴情報の伝送のために、図78に示すようなRe_Coding information Bus macroblock formatが規定される。このマクロブロックは、16×16(=256)ビットで構成される。そして、そのうちの図78において上から3行目と4行目に示す32ビットが、picrate_elementとされる。このpicrate_elementには、図79乃至図81に示すPicture rate elementsが記述される。図79の上から2行目に1ビットのred_bw_flagが規定されてお

り、また、3行目に3ビットのred_bw_indicatorが規定されている。即ち、これらのフラグred_bw_flag, red_bw_indicatorは、図78のpicrate_elementとして伝送される。

【0383】図78のその他のデータについて説明すると、SRIB_sync_codeは、このフォーマットのマクロブロックの最初の行が左詰めにアライメントされていることを表すコードであり、具体的には、“11111”に設定される。fr_fl_SRIBは、picture_structureがフレームピクチャ構造の場合(その値が“11”である場合)、1に設定され、Re_Coding Information Bus macroblockが16ラインを超えて伝送されることを表し、picture_structureがフレーム構造ではない場合、0に設定され、Re_Coding Information Busが16ラインを超えて伝送されることを意味する。この機構により、Re_Coding Information Busが、空間的かつ時間的にデコードされたビデオフレームまたはフィールドの対応する画素にロックされる。

【0384】SRIB_top_field_firstは、元のビットストリームに保持されているtop_field_firstと同じ値に設定され、関連するビデオのRe_Coding Information Busの時間的アライメントをrepeat_first_fieldとともに表している。SRIB_repeat_first_fieldは、元のビットストリームに保持されているrepeat_first_fieldと同じ値に設定される。first fieldのRe_Coding Information Busの内容は、このフラグに示されるように繰り返される必要がある。

【0385】422_420_chromaは、元のビットストリームが4:2:2または4:2:0のいずれであるかを表す。その値の0は、ビットストリームが4:2:0であり、色差信号のアップサンプリングが、4:2:2のビデオが出力されるように行われたことを表す。その値の0は、色差信号のフィルタリング処理が実行されていないことを表す。

【0386】rolling_SRIB_mb_refは、16ビットのモジュロ65521を表し、この値は、毎マクロブロック毎にインクリメントされる。この値は、フレームピクチャ構造のフレームに渡って連続している必要がある。さもなくば、この値は、フィールドに渡って連続している必要がある。この値は、0から65520の間の所定の値に初期化される。これにより、レコーダのシステムに、ユニークなRe_Coding Information Busの識別子を組み込むことが許容される。

【0387】Re_Coding Information Bus macroblockのその他のデータの意味は、上述した通りであるので、ここでは省略する。

【0388】図82に示すように、図78の256ビットのRe_Coding Information Busのデータは、1ビットずつ、色差データのLSBであるCb[0][0], Cr[0][0], Cb[1][0], Cr[1][0]に配置される。図82に示すフォーマ

ットにより、4ビットのデータを送ることができるので、図78の256ビットのデータは、図82のフォーマットを64(=256/4)個送ることによって伝送することができる。

【0389】なお、上記各処理を行うコンピュータプログラムは、磁気ディスク、CD-ROM等の情報記録媒体よりなる提供媒体のほか、インターネット、デジタル衛星などのネットワーク提供媒体を介してユーザに提供することができる。

【0390】

【発明の効果】以上の如く、請求項1に記載のストリーム伝送装置、請求項4に記載のストリーム伝送方法、および請求項5に記載の提供媒体によれば、過去の符号化処理において使用された複数の符号化パラメータを、選択的に組み合わせて、その符号化履歴情報を生成し、符号化ストリームに重乗するようにしたので、再符号化に伴う画像の劣化を抑制可能なストリームを、少ない容量のメディアを介して伝送することが可能となる。

【図面の簡単な説明】

【図1】従来のトランスコーダ131の構成の例を示すブロック図である。

【図2】従来のトランスコーダ131の他の構成の例を示すブロック図である。

【図3】従来の符号化装置と復号装置の配置を説明する図である。

【図4】高効率符号化の原理を説明する図である。

【図5】画像データを圧縮する場合におけるピクチャタイプを説明する図である。

【図6】画像データを圧縮する場合におけるピクチャタイプを説明する図である。

【図7】動画像信号を符号化する原理を説明する図である。

【図8】動画像信号を符号化し、復号する装置の構成を示すブロック図である。

【図9】画像データの構成を説明する図である。

【図10】図8のエンコーダ18の構成を示すブロック図である。

【図11】図10の予測モード切替回路52の動作を説明する図である。

【図12】図10の予測モード切替回路52の動作を説明する図である。

【図13】図10の予測モード切替回路52の動作を説明する図である。

【図14】図10の予測モード切替回路52の動作を説明する図である。

【図15】図8のデコーダ31の構成を示すブロック図である。

【図16】ピクチャタイプに対応したSNR制御を説明する図である。

【図17】本発明を適用したトランスコーダ101の構

成を示すブロック図である。

【図18】図17のトランスコーダ101のより詳細な構成を示すブロック図である。

【図19】図17の復号装置102に内蔵されるデコーダ111の構成を示すブロック図である。

【図20】マクロブロックの画素を説明する図である。

【図21】符号化パラメータが記録される領域を説明する図である。

【図22】図17の符号化装置106に内蔵されるエンコーダ121の構成を示すブロック図である。

【図23】図18のヒストリーフォーマッタ211の構成例を示すブロック図である。

【図24】図18のヒストリーデコーダ203の構成例を示すブロック図である。

【図25】図18のコンバータ212の構成例を示すブロック図である。

【図26】図25のスタンプ回路323の構成例を示すブロック図である。

【図27】図25のコンバータ212の動作を説明するタイミングチャートである。

【図28】図18のコンバータ202の構成例を示すブロック図である。

【図29】図28のディリット回路343の構成例を示すブロック図である。

【図30】図18のコンバータ212の他の構成例を示すブロック図である。

【図31】図18のコンバータ202の他の構成例を示すブロック図である。

【図32】図18のユーザデータフォーマッタ213の構成例を示すブロック図である。

【図33】図17のトランスコーダ101が実際に使用される状態を示す図である。

【図34】符号化パラメータが記録される領域を説明する図である。

【図35】図17の符号化装置106の変更可能ピクチャタイプ判定処理を説明するフローチャートである。

【図36】ピクチャタイプが変更される例を示す図である。

【図37】ピクチャタイプが変更される他の例を示す図である。

【図38】図17の符号化装置106の量子化制御処理を説明する図である。

【図39】図17の符号化装置106の量子化制御処理を説明するフローチャートである。

【図40】密結合されたトランスコーダ101の構成を示すブロック図である。

【図41】ビデオシーケンスのストリームのシンタックスを説明する図である。

【図42】図41のシンタックスの構成を説明する図である。

【図 4 3】固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 4】固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 5】固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 6】固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 7】固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 8】固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 4 9】固定長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 5 0】可変長の履歴情報を記録するhistory_stream()のシンタックスを説明する図である。

【図 5 1】sequence_header()のシンタックスを説明する図である。

【図 5 2】sequence_extension()のシンタックスを説明する図である。

【図 5 3】extension_and_user_data()のシンタックスを説明する図である。

【図 5 4】user_data()のシンタックスを説明する図である。

【図 5 5】group_of_pictures_header()のシンタックスを説明する図である。

【図 5 6】picture_header()のシンタックスを説明する図である。

【図 5 7】picture_coding_extension()のシンタックスを説明する図である。

【図 5 8】extension_data()のシンタックスを説明する図である。

【図 5 9】sequence_display_extension()のシンタックスを説明する図である。

【図 6 0】quant_matrix_extension()のシンタックスを説明する図である。

【図 6 1】copyright_extension()のシンタックスを説明する図である。

【図 6 2】picture_display_extension()のシンタックスを説明する図である。

【図 6 3】picture_data()のシンタックスを説明する図である。

【図 6 4】slice()のシンタックスを説明する図である。

【図 6 5】macroblock()のシンタックスを説明する図である。

【図 6 6】macroblock_modes()のシンタックスを説明する図である。

【図 6 7】I ピクチャに対するmacroblock_typeの可変長符号を説明する図である。

【図 6 8】P ピクチャに対するmacroblock_typeの可変長符号を説明する図である。

【図 6 9】B ピクチャに対するmacroblock_typeの可変長符号を説明する図である。

【図 7 0】motion_vectors(s)のシンタックスを説明する図である。

【図 7 1】motion_vector(r, s)のシンタックスを説明する図である。

【図 7 2】coded_block_pattern()のシンタックスを説明する図である。

【図 7 3】本発明を適用したトランスコード 101 の他の構成を示すブロック図である。

【図 7 4】履歴情報の項目の組み合わせを説明する図である。

【図 7 5】図 7 3 のトランスコード 101 の動作を説明するフローチャートである。

【図 7 6】re_coding_stream_info()のシンタックスを説明する図である。

【図 7 7】red_bw_flag, red_bw_indicatorを説明する図である。

【図 7 8】Re_Coding Information Bus macroblock formationを説明する図である。

【図 7 9】Picture rate elementsを説明する図である。

【図 8 0】Picture rate elementsを説明する図である。

【図 8 1】Picture rate elementsを説明する図である。

【図 8 2】Re_Coding Information Busが記録される領域を説明する図である。

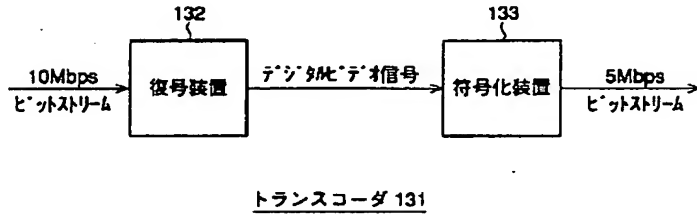
【符号の説明】

1 符号化装置, 2 復号化装置, 3 記録媒体,
12, 13 A/D変換器, 14 フレームメモリ,
15 輝度信号フレームメモリ, 16 色差信号フレームメモリ,
17 フォーマット変換回路, 18 エンコーダ, 31 デコーダ, 32 フォーマット変換回路,
33 フレームメモリ, 34 輝度信号フレームメモリ, 35 色差信号フレームメモリ, 36, 37 D/A変換器, 50 動きベクトル検出回路,
51 フレームメモリ, 52 予測モード切り替え回路, 53 演算部, 54 予測判定回路, 55 DCTモード切り替え回路, 56 DCT回路, 57 量子化回路, 58 可変長符号化回路, 59 送信バッファ, 60 逆量子化回路, 61 IDCT回路, 62 演算器, 63 フレームメモリ, 64 動き補償回路, 81 受信バッファ, 82 可変長復号化回路, 83 逆量子化回路, 84 IDCT回路, 85 演算器, 86 フレームメモリ, 87 動き補償回路, 101 トランスコード, 102 復号装置, 103 符号化パラメータ多重装置, 105 符号

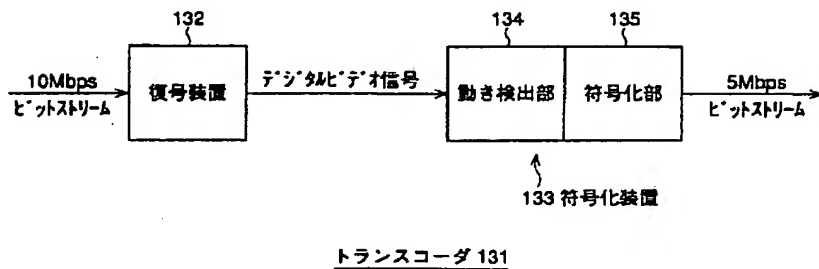
71

化パラメータ分離装置, 106 符号化装置, 106 SDTI, 111 デコーダ, 112 可変長復号化回路, 121 エンコーダ, 122 符号化パラ

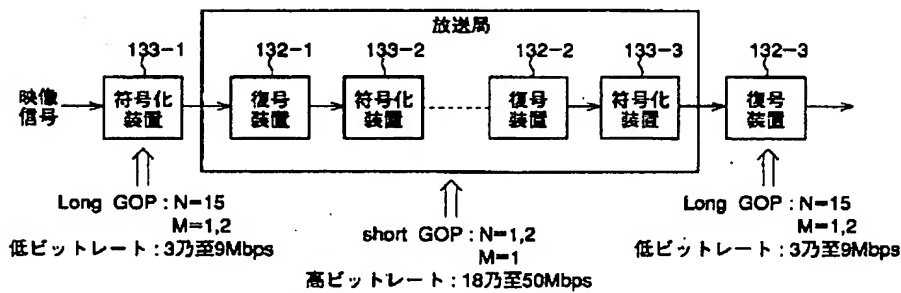
【図1】



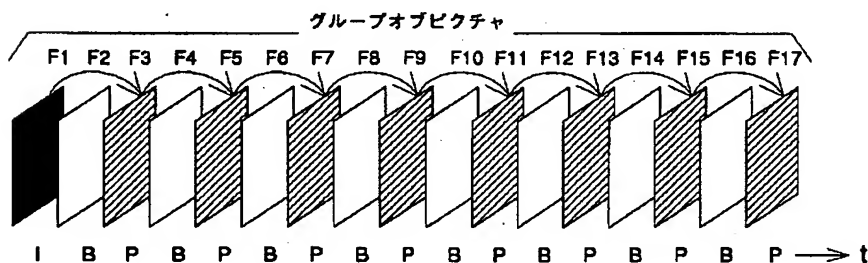
【図2】



【図3】



【図5】



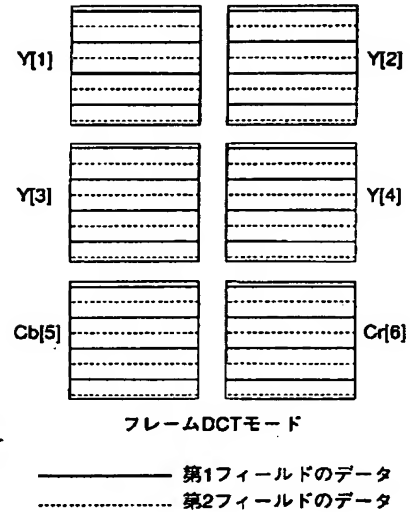
前方予測 P-ピクチャ

ピクチャタイプ I,P,B-picture

72

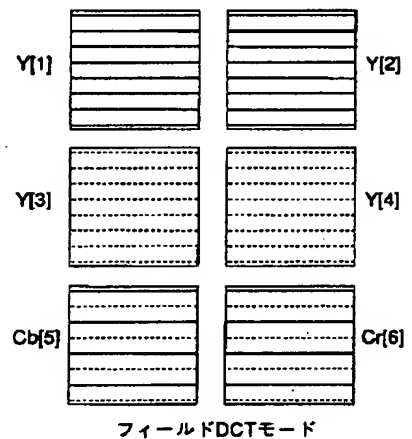
メータコントローラ, 131 トランスコーダ, 132 復号装置, 133 符号化装置, 134 動き検出部, 135 符号化部

【図13】



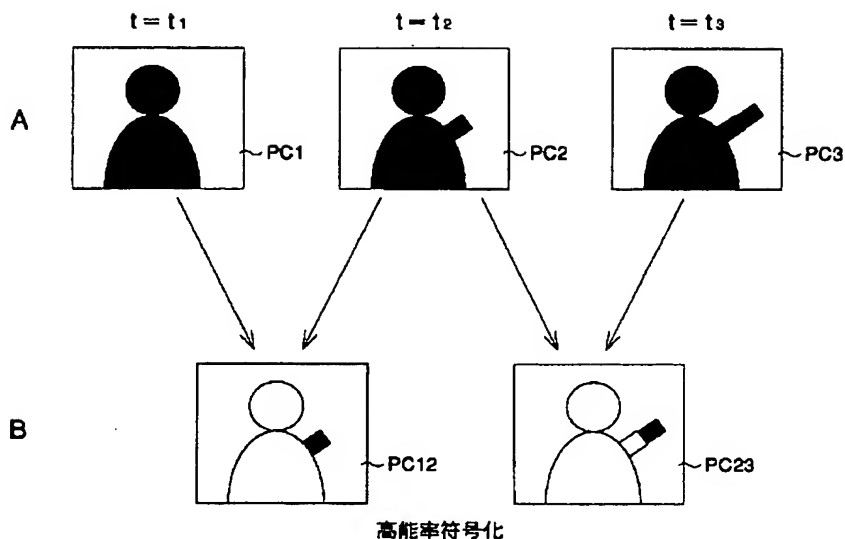
フレーム/フィールドDCTモード

【図14】

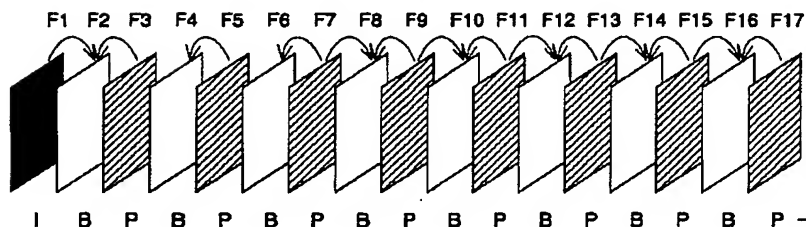


フレーム/フィールドDCTモード

【図 4】



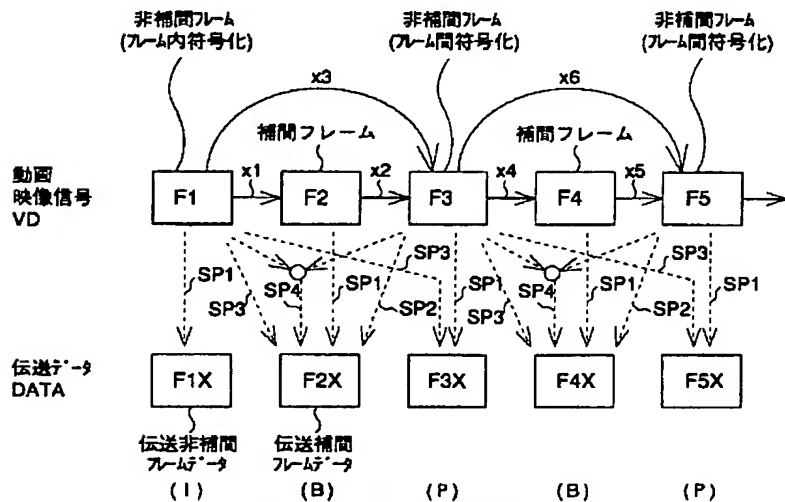
【図 6】



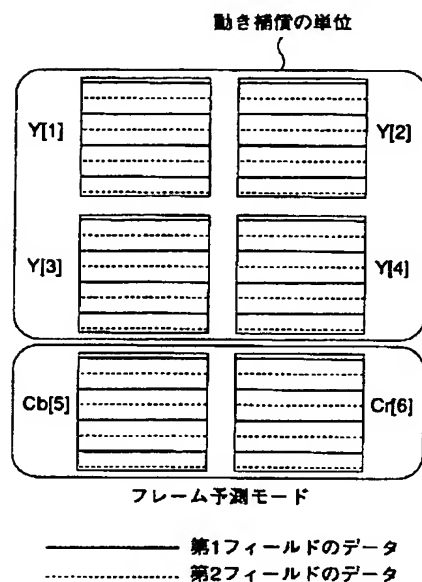
両方向予測 B-ピクチャ

ピクチャタイプ I,P,B-picture

【図 7】



【图 1-1】



フレイム/フィールド予測モード

【图 77】

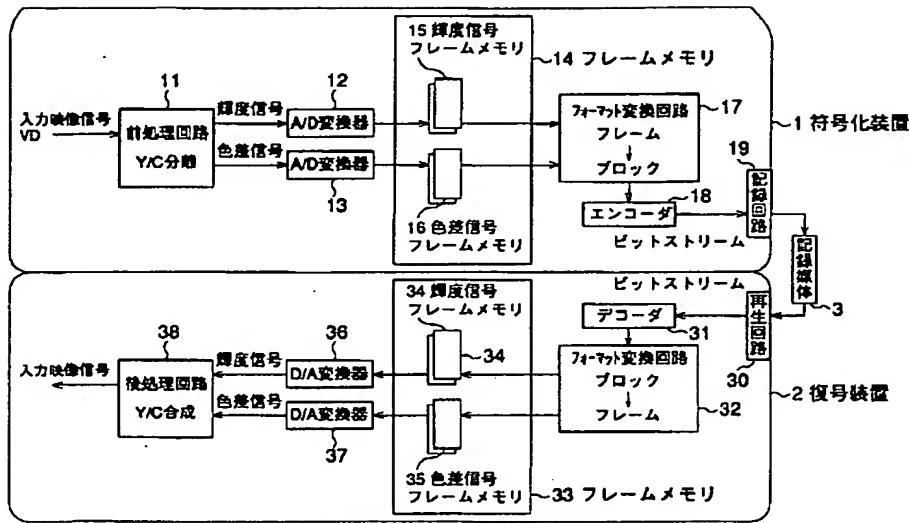
項目の組合せ	red_bw_flag	
	red_bw_indicator	
組合せ1	0	-
組合せ2	1	0
組合せ3	1	1
組合せ4	1	2
組合せ5	1	3

【圖 20】

0	1	2	13	14	15
16	17	18	29	30	31
32	33	34	45	46	47
.
.
.
.
.
.
.
208	209	210	221	222	223
224	225	226	237	238	239
240	241	242	251	254	255

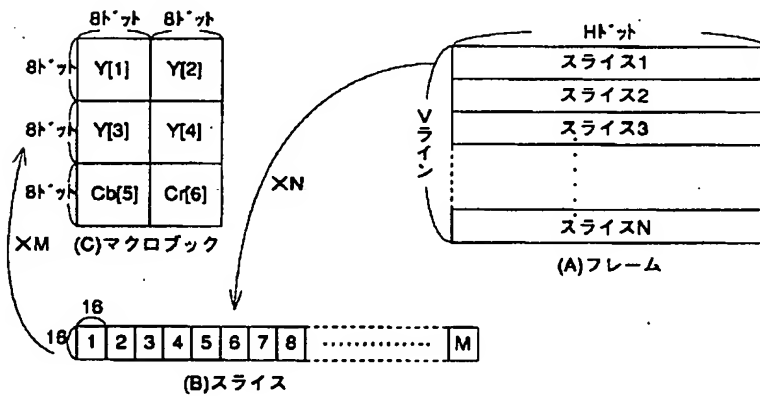
マクロブロック

【図8】



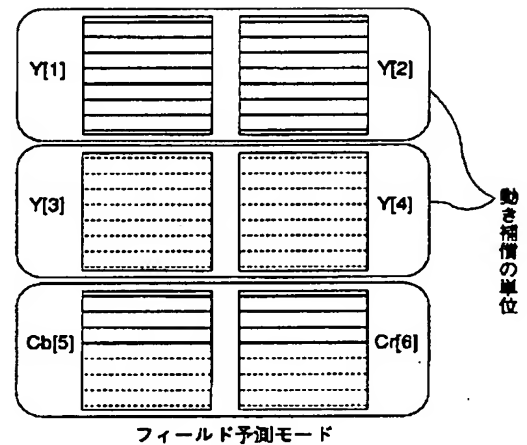
動画画像符号化/復号化装置

【図9】



画像データの構造

【図12】

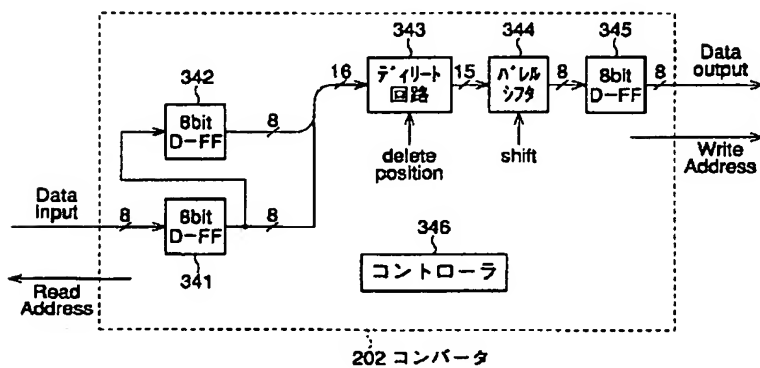


フィールド予測モード

—— 第1フィールドのデータ
 第2フィールドのデータ

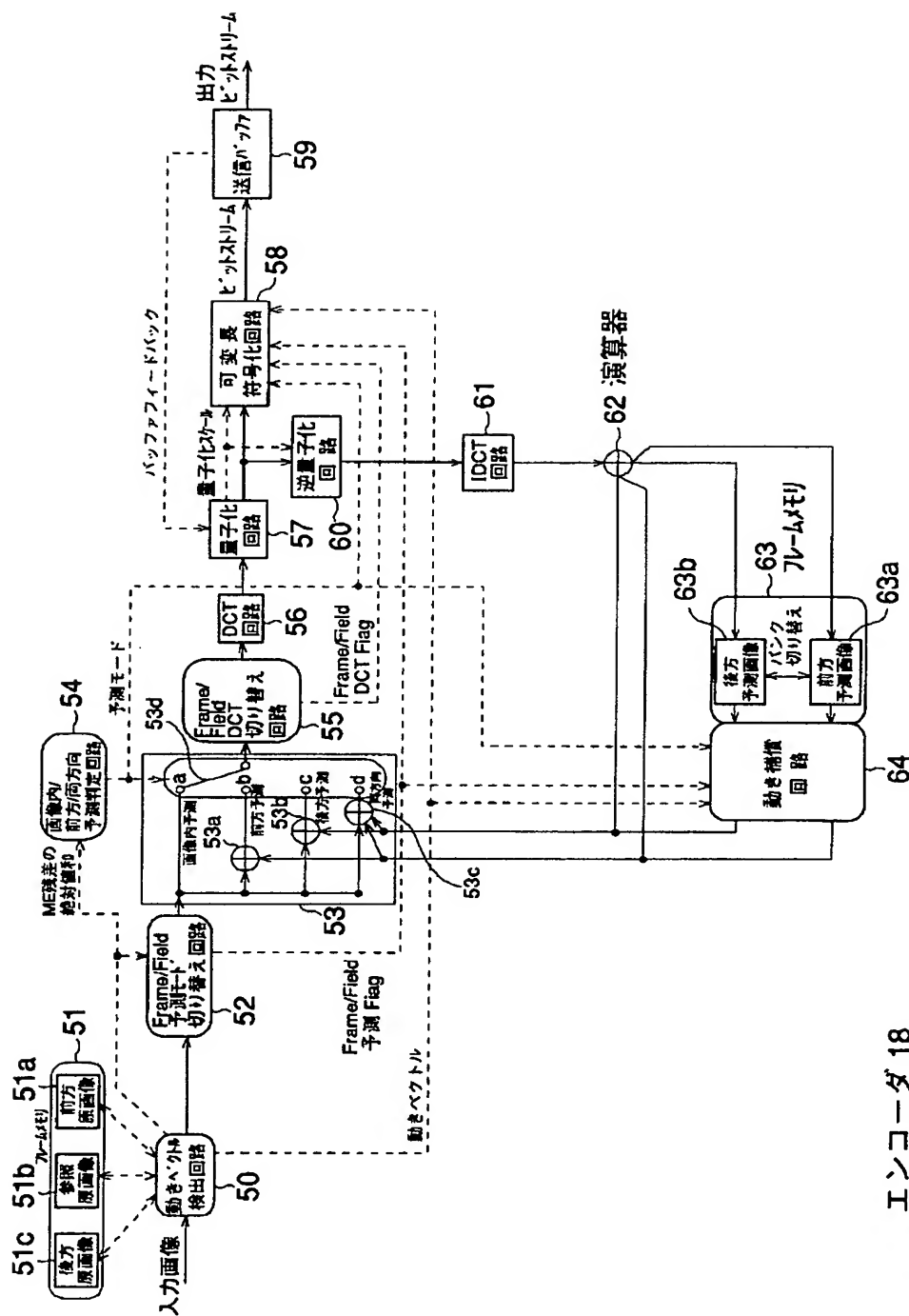
フレーム/フィールド予測モード

【図28】

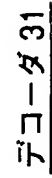


202 コンバータ

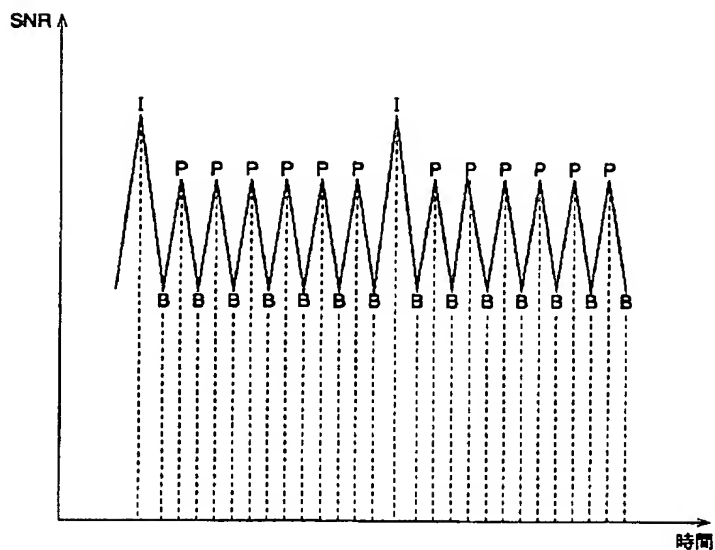
18-ダ-コ-ン



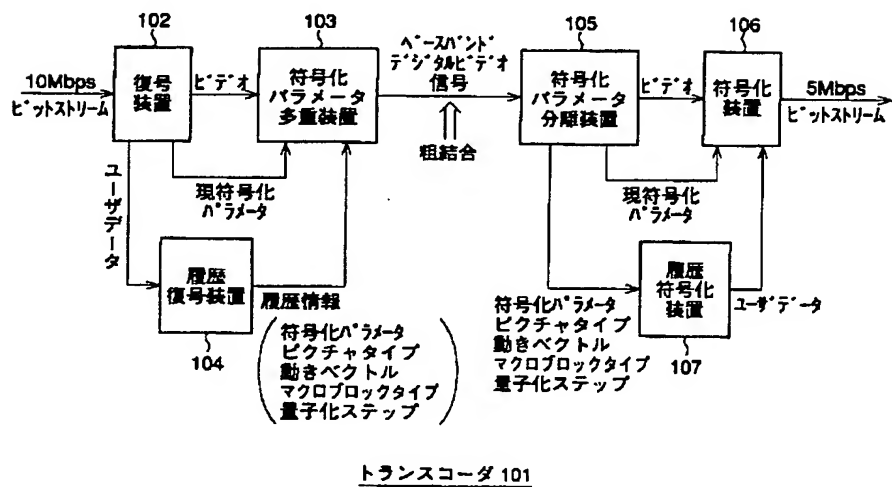
90 復号回路



【図 16】



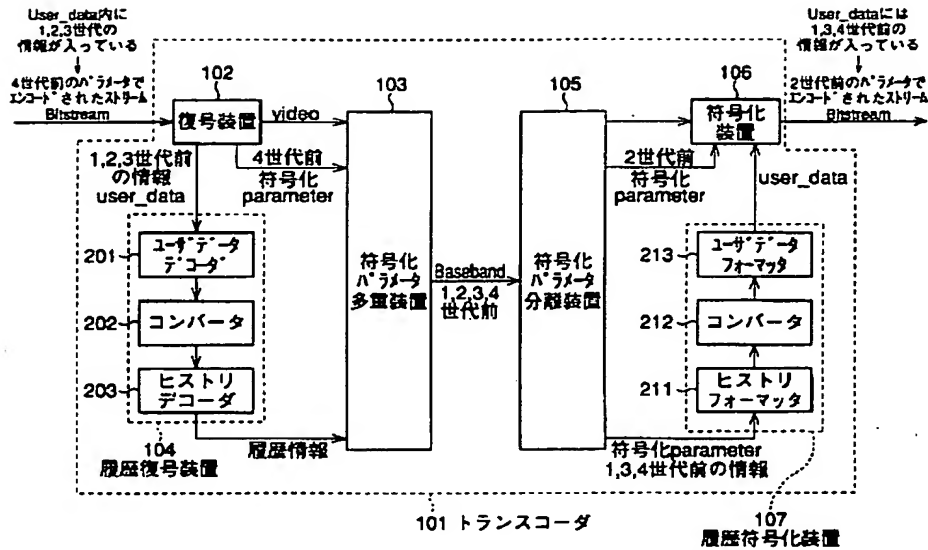
【図 17】



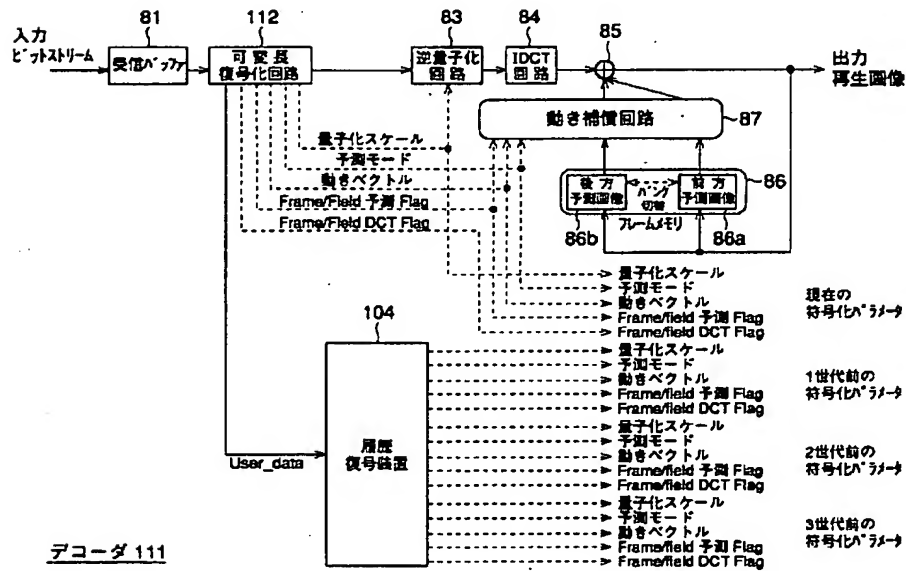
【図 21】

区間										画像データ領域
D9	Cb[0][9]	Y[0][9]	Cr[0][9]	Y[1][9]	Cb[1][9]	Y[2][9]	Cr[1][9]	Y[3][9]		
D8	Cb[0][8]	Y[0][8]	Cr[0][8]	Y[1][8]	Cb[1][8]	Y[2][8]	Cr[1][8]	Y[3][8]		
D7	Cb[0][7]	Y[0][7]	Cr[0][7]	Y[1][7]	Cb[1][7]	Y[2][7]	Cr[1][7]	Y[3][7]		
D6	Cb[0][6]	Y[0][6]	Cr[0][6]	Y[1][6]	Cb[1][6]	Y[2][6]	Cr[1][6]	Y[3][6]		
D5	Cb[0][5]	Y[0][5]	Cr[0][5]	Y[1][5]	Cb[1][5]	Y[2][5]	Cr[1][5]	Y[3][5]		
D4	Cb[0][4]	Y[0][4]	Cr[0][4]	Y[1][4]	Cb[1][4]	Y[2][4]	Cr[1][4]	Y[3][4]		
D3	Cb[0][3]	Y[0][3]	Cr[0][3]	Y[1][3]	Cb[1][3]	Y[2][3]	Cr[1][3]	Y[3][3]		
D2	Cb[0][2]	Y[0][2]	Cr[0][2]	Y[1][2]	Cb[1][2]	Y[2][2]	Cr[1][2]	Y[3][2]		
D1	1世代前		2世代前		3世代前		最新			
D0										
	Cb[0][x]	Y[0][x]	Cr[0][x]	Y[1][x]	Cb[1][x]	Y[2][x]	Cr[1][x]	Y[3][x]	符号化パラメータ領域	

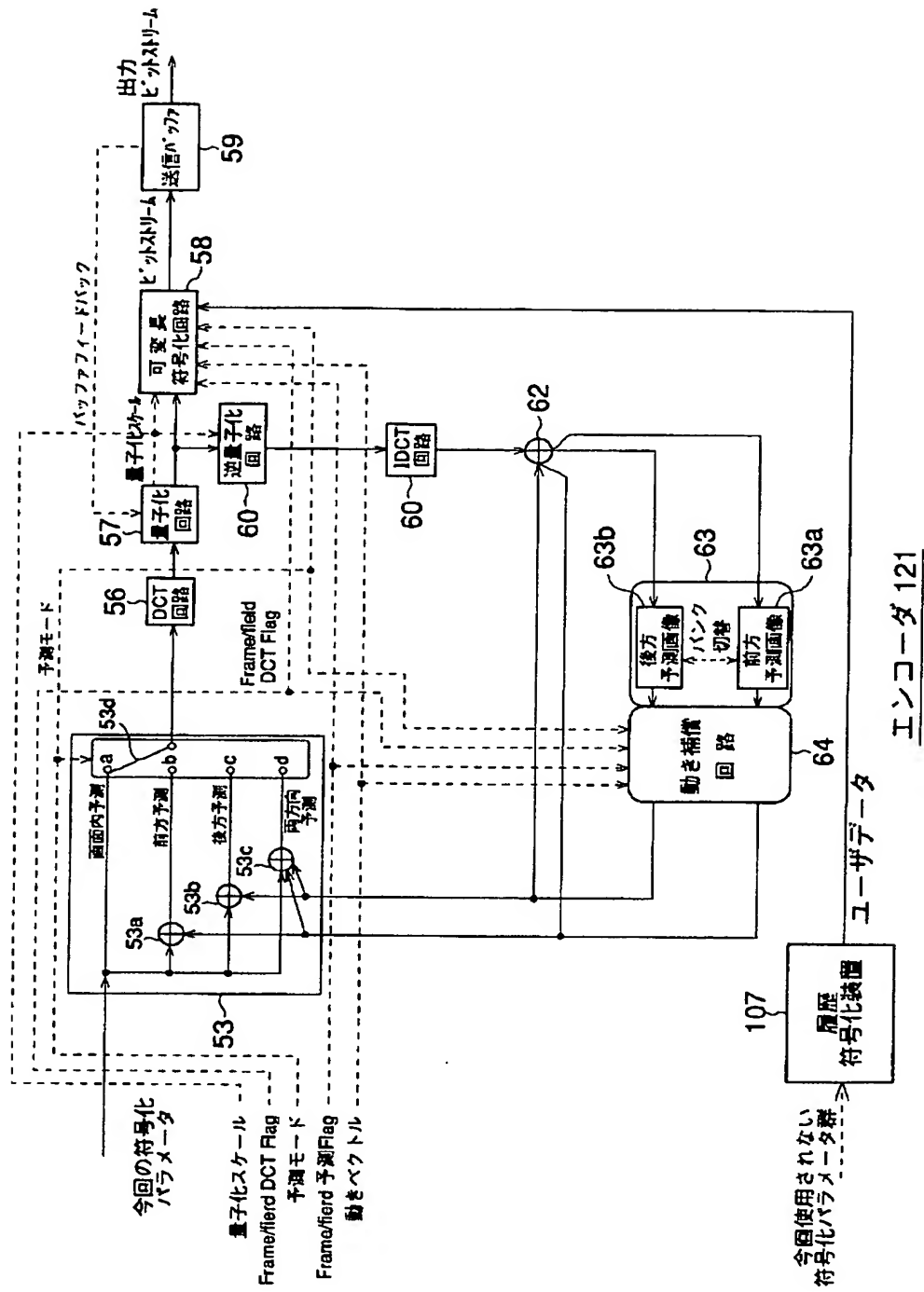
【図 18】



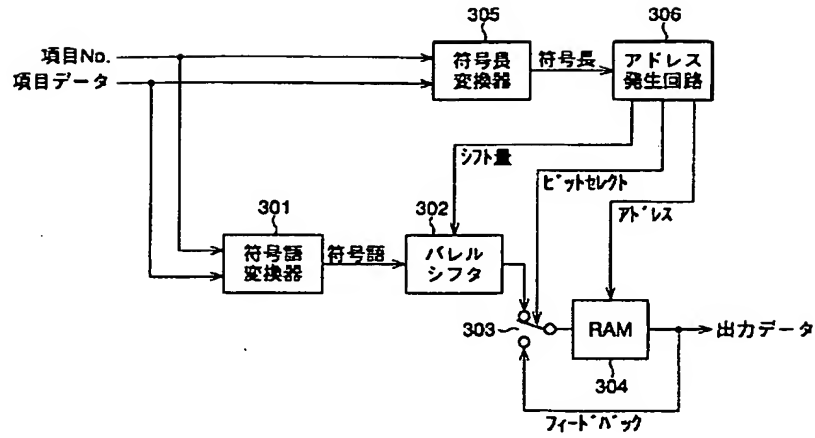
【図 19】



【図 22】

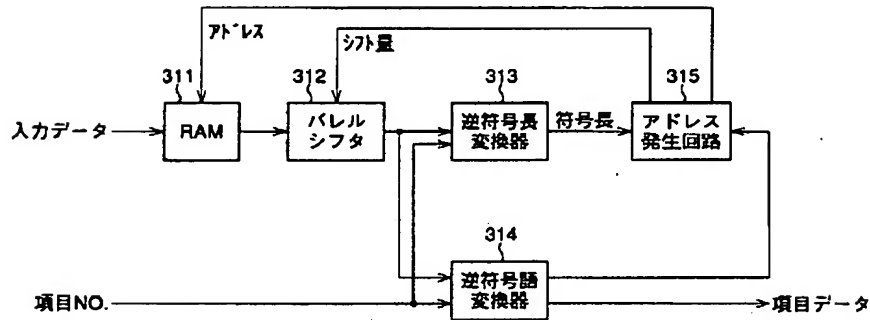


【図 23】



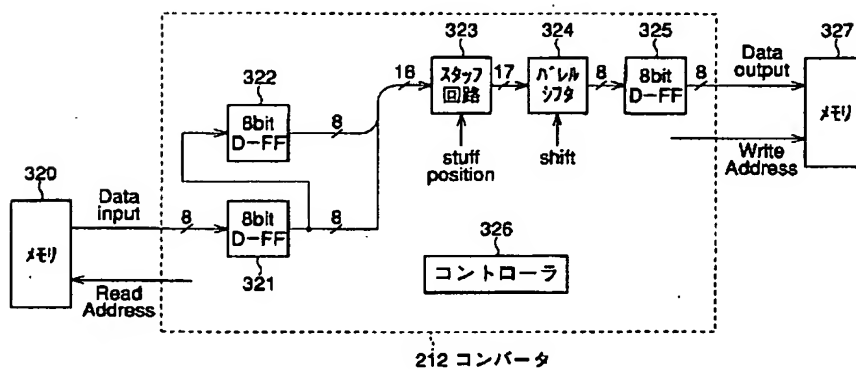
ヒストリフォーマッタ 211

【図 24】



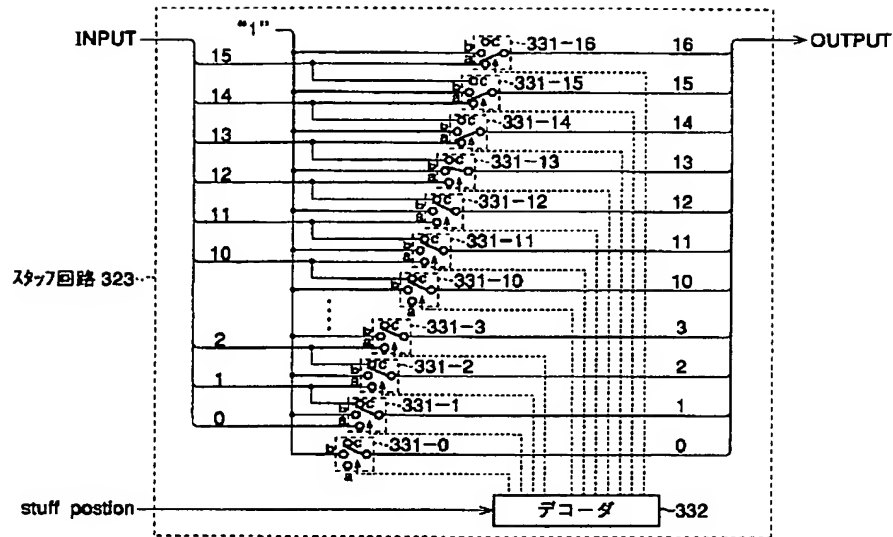
ヒストリデコーダ 203

【図 25】

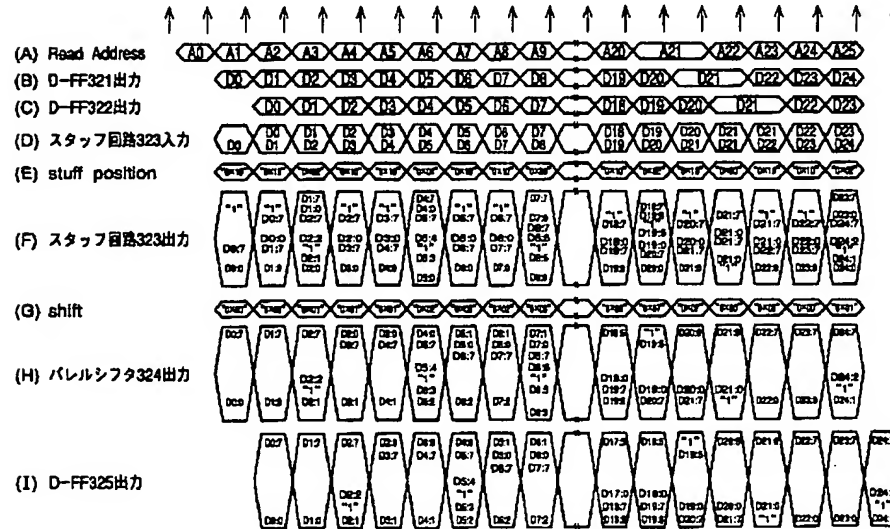


212 コンバータ

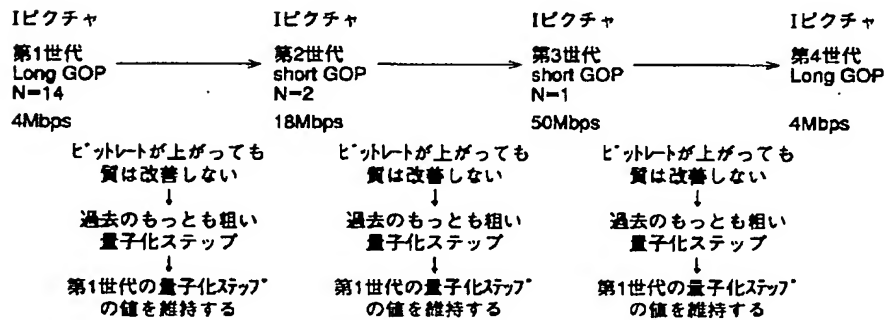
【図 26】



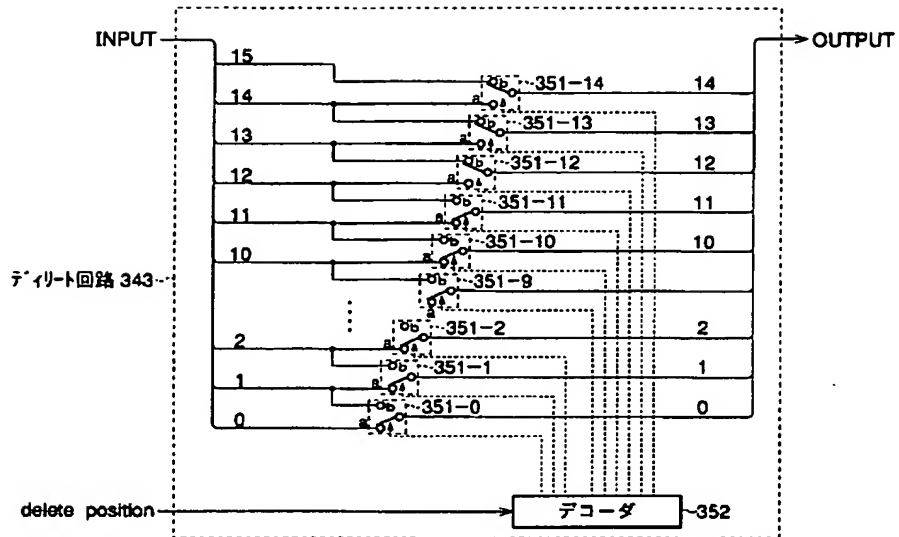
【図 27】



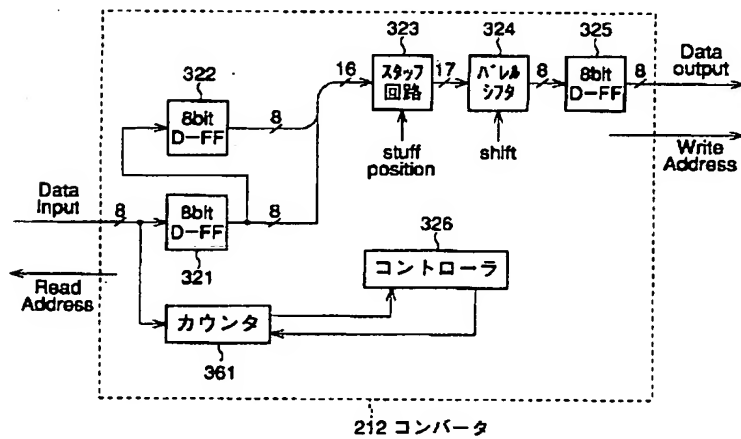
【図 38】



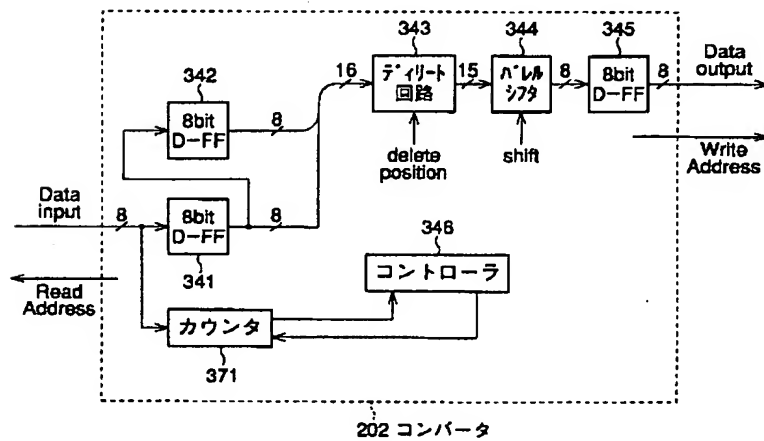
【図 29】



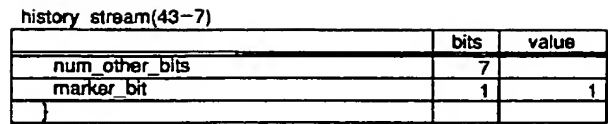
【図 30】



【図 31】






【図 49】



D9								
D8								
D7								
D6								
D5								
D4								
D3								
D2								
D1								
D0								
	Cb[0][x]	Y[0][x]	Cr[0][x]	Y[1][x]	Cb[1][x]	Y[2][x]	Cr[1][x]	Y[3][x]

画像データ領域

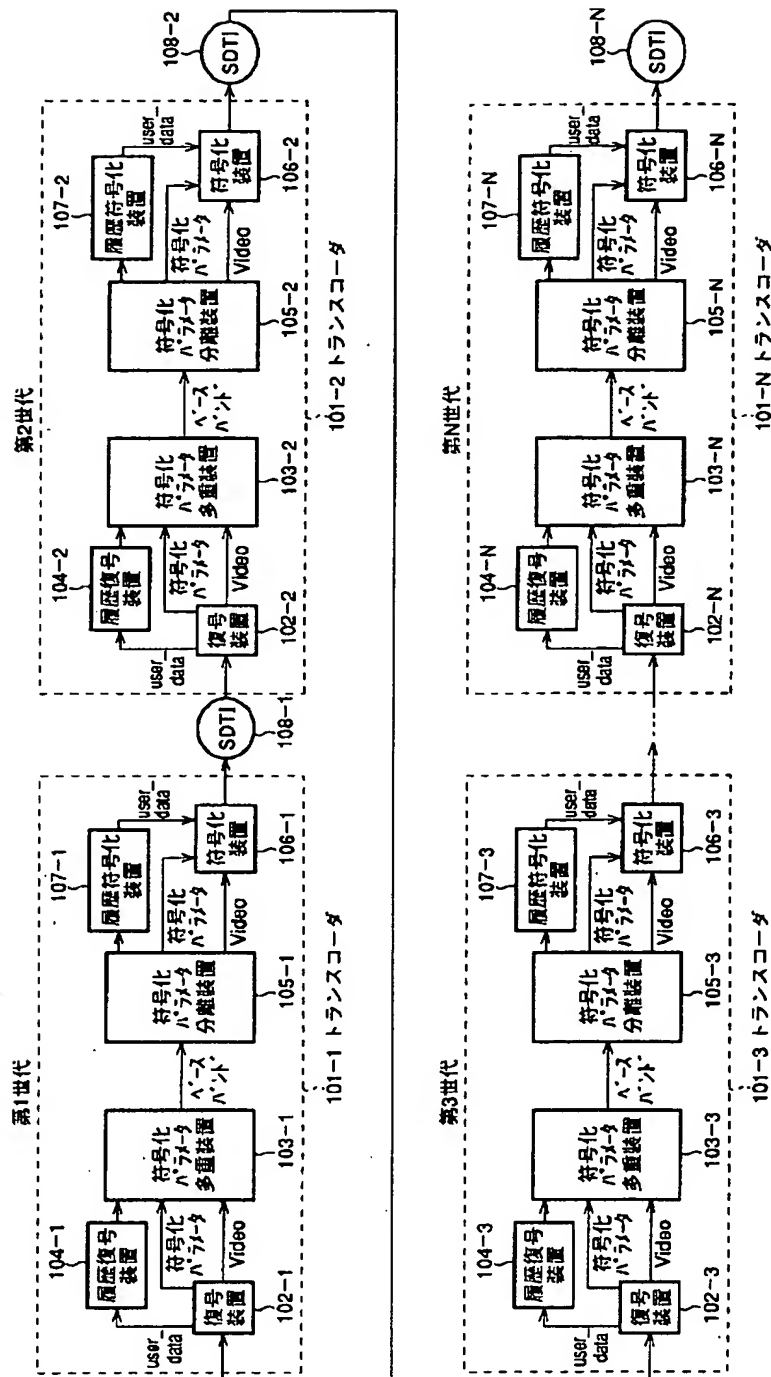
符号化ハッシュ領域

	Iピクチャの符号化パラメータ
	Pピクチャの符号化パラメータ
	Bピクチャの符号化パラメータ

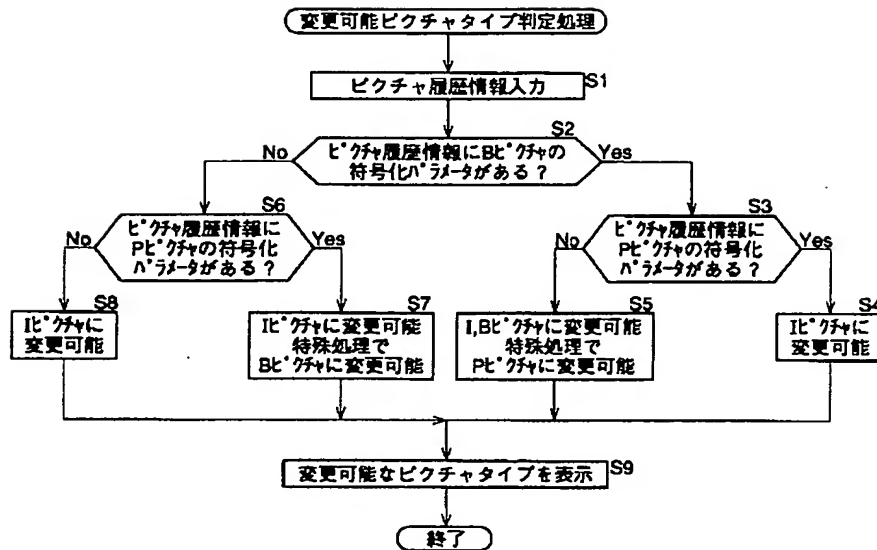
The diagram illustrates the structure of GOP (Group of Pictures) for four different video coding scenarios. Each scenario shows a sequence of frames (B, P, I) and their corresponding bitstream representations (dots and vertical bars).

- 第1世代 Long GOP N=14**: Shows a sequence of frames starting with B, followed by P, B, P, B, P, B, P, B, P, B, P, B, P, B, P. The bitstream representation shows a long sequence of dots and vertical bars, indicating a long GOP structure.
- 第2世代 short GOP N=2**: Shows a sequence of frames starting with B, followed by I, B, I, B, I, B, I, B, I, B, I, B, I, B, I. The bitstream representation shows a shorter sequence of dots and vertical bars, indicating a short GOP structure.
- 第3世代 short GOP N=1**: Shows a sequence of frames starting with I, followed by I, I, I, I, I, I, I, I, I, I, I, I, I, I, I. The bitstream representation shows a very short sequence of dots and vertical bars, indicating a short GOP structure.
- 第4世代 Random GOP N=?**: Shows a sequence of frames starting with B, followed by P, B, P, B, P, B, I, B, P, B, P, B, B, P, B, I, B, P, B, P, B, P. The bitstream representation shows a sequence of dots and vertical bars, indicating a random GOP structure.

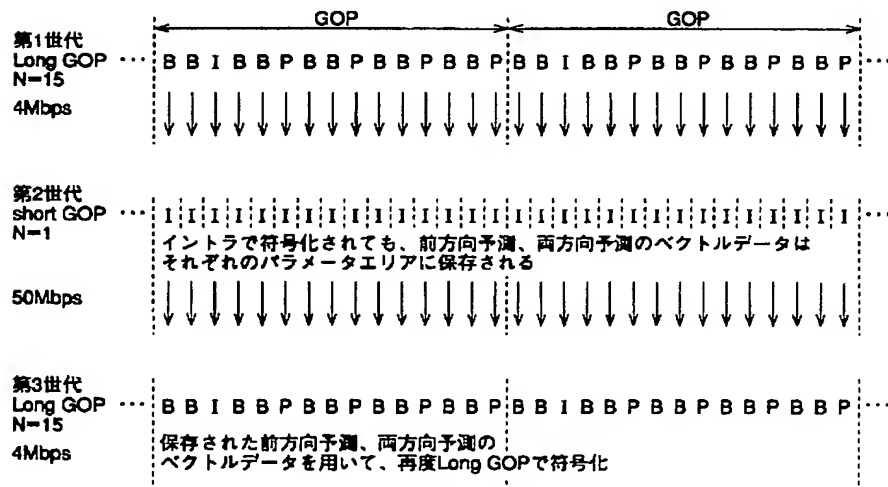
【図 33】



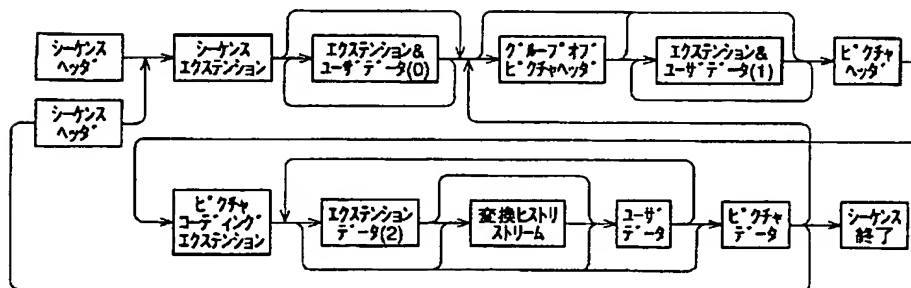
【図35】



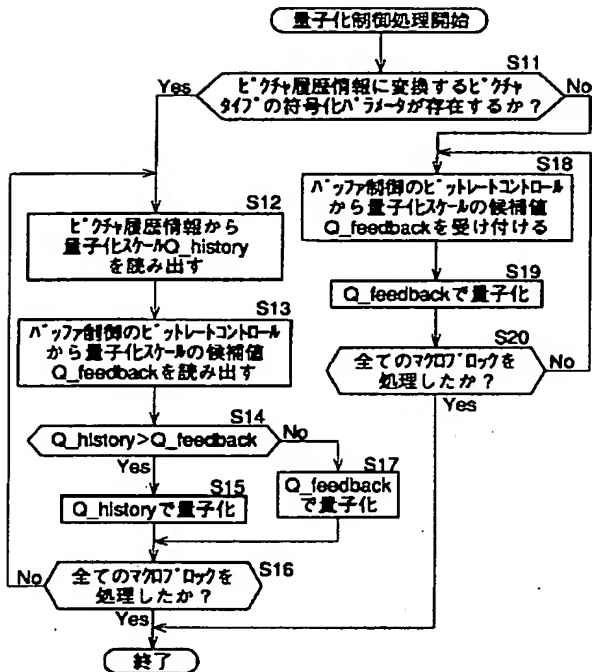
【図36】



【図42】



【図 39】



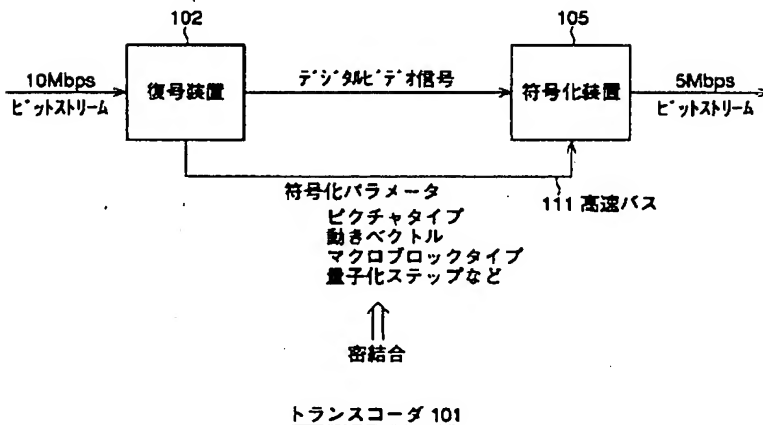
【図 50】

history_stream()	No. of bits	Mnemonic
next_start_code()		
sequence_header()		
sequence_extension()		
extension_and_user_data(0)		
if(nextbits() == group_start_code){		
group_of_pictures_header()		
extension_and_user_data(1)		
}		
picture_header()		
picture_coding_extension()		
re_coding_stream_info()		
extension_and_user_data(2)		
if(!red_bw_flag (red_bw_indicator < 2))		
picture_data()		
sequence_end_code	32	bslbf

【図 55】

group_of_pictures_header()	No. of bits	Mnemonic
group_start_code	32	bslbf
time_code	25	bslbf
closed_gop	1	uimbsbf
broken_link	1	uimbsbf
next_start_code()		

【図 40】



【図 53】

extension_and_user_data()	No. of bits	Mnemonic
while((nextbits() == extension_start_code)		
(nextbits() == user_data_start_code))		
if((i-1) && (nextbits() == extension_start_code))		
extension_data(i)		
if(nextbits() == user_data_start_code)		
user_data()		

【図 54】

user_data()	No. of bits	Mnemonic
user_data_start_code	32	bslbf
while((nextbits() != '0000 0000 0000 0000 0001'))		
user_data()	8	uimbsbf
next_start_code()		

【図 4 1】

stream with history data		
video_sequence()	No. of bits	Mnemonic
next_start_code()		
sequence_header()		
sequence_extension()		
do{		
extension_and_user_data(0)		
do{		
if(nextbits() == group_start_code){		
group_of_pictures_header(1)		
extension_and_user_data(1)		
}		
picture_header()		
picture_coding_extension()		
while((nextbits() == extension_start_code)		
(nextbits() == user_data_start_code)){		
if(nextbits() == extension_start_code){		
extension_data(2)		
if(nextbits() == user_data_start_code){		
user_data_start_code	32	bslbf
if(nextbits() == History_Data_ID){		
History_Data_ID	32	bslbf
converted_history_stream()		
}		
} else{		
user_data()		
}		
}		
picture_data()		
}while((nextbits() == picture_start_code)		
(nextbits() == group_start_code))		
if(nextbits() == sequence_end_code){		
sequence_header()		
sequence_extension()		
}		
}while(nextbits() != sequence_end_code)		
sequence_end_code	32	bslbf
}		

【図 4 3】

history stream(43-1)		
history_stream()	bits	value
sequence_header		
sequence_header_code	32	000001B3
sequence_header_present_flag	1	
horizontal_size_value	12	
marker_bit	1	1
vertical_size_value	12	
aspect_ratio_information	4	
frame_rate_code	4	
marker_bit	1	1
bit_rate_value	18	
marker_bit	1	1
vbv_buffer_size_value	10	
constrained_parameter_flag	1	0
load_intra_quantiser_matrix	1	
load_non_intra_quantiser_matrix	1	
marker_bits	5	1F
intra_quantiser_matrix[64]	8*64	
non_intra_quantiser_matrix[64]	8*64	
sequence_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	1
sequence_extension_present_flag	1	
profile_and_level_indication	8	
progressive_sequence	1	
chroma_format	2	
horizontal_size_extension	2	
vertical_size_extension	2	
marker_bit	1	1
bit_rate_extension	12	
vbv_buffer_size_extension	8	
low_delay	1	
marker_bit	1	1

【図 4 5】

【図 5 1】

sequence_header()		
	No. of bits	Mnemonic
sequence_header_code	32	bslbf
horizontal_size_value	12	uimsbf
vertical_size_value	12	uimsbf
aspect_ratio_information	4	uimsbf
frame_rate_code	4	uimsbf
bit_rate_value	18	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_value	10	uimsbf
constrained_parameters_flag	1	bslbf
load_intra_quantiser_matrix	1	uimsbf
if(load_intra_quantiser_matrix)		
intra_quantiser_matrix[64]	8*64	uimsbf
load_non_intra_quantiser_matrix	1	uimsbf
if(load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix[64]	8*64	uimsbf
next_start_code()		

history stream(43-3)

	bits	value
temporal_reference	10	
picture_coding_type	3	
marker_bit	1	1
vbv_delay	16	
full_pel_forward_vector	1	
forward_f_code	3	
full_pel_backward_vector	1	
marker_bit	1	1
backward_f_code	3	
marker_bit	1	1
picture_coding_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	8
f_code[0][0]	4	
f_code[0][1]	4	
f_code[1][0]	4	
f_code[1][1]	4	
intra_dc_precision	2	
picture_structure	2	
top_field_first	1	
frame_pred_frame_dct	1	
concealment_motion_vectors	1	
q_scale_type	1	
marker_bit	1	1
intra_vlc_format	1	
alternate_scan	1	
repeat_first_field	1	
chroma_420_type	1	
progressive_frame	1	
composite_display_flag	1	
v_axis	1	
field_sequence	3	
sub_carrier	1	
burst_amplitude	7	

【図 4 4】

history stream(43-2)

	bits	value
frame_rate_extension_n	2	
frame_rate_extension_d	5	
marker_bits	6	3F
sequence_display_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	2
sequence_display_extension_present_flag	1	
video_format	3	
colour_description	1	
colour_primaries	8	
transfer_characteristics	8	
marker_bit	1	1
matrix_coefficients	8	
display_horizontal_size	14	
marker_bit	1	1
display_vertical_size	14	
marker_bit	1	1
macroblock_assignment_in_user_data		
macroblock_assignment_present_flag	1	
marker_bits	7	7F
v_phase	8	
h_phase	8	
group_of_picture_header		
group_start_code	32	000001B8
group_of_picture_header_present_flag	1	
time_code	25	
closed_gop	1	
broken_link	1	
marker_bits	4	F
picture_header		
picture_start_code	32	00000100

【図 4 6】

history stream(43-4)

	bits	value
marker_bit	1	1
sub_carrier_phase	8	
quant_matrix_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	3
quant_matrix_extension_present_flag	1	
load_intra_quantiser_matrix	1	
marker_bits	2	3
intra_quantiser_matrix[64]	8*64	
load_non_intra_quantiser_matrix	1	
marker_bits	7	7F
non_intra_quantiser_matrix[64]	8*64	
load_chroma_intra_quantiser_matrix	1	
marker_bits	7	7F
chroma_intra_quantiser_matrix[64]	8*64	
load_chroma_non_intra_quantiser_matrix	1	
marker_bits	7	7F
chroma_non_intra_quantiser_matrix[64]	8*64	
copyright_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	4
copyright_extension_present_flag	1	
copyright_flag	1	
copyright_identifier	8	
original_or_copy	1	
marker_bit	1	
copyright_number_1	20	
marker_bit	1	
copyright_number_2	22	
marker_bit	1	
copyright_number_3	22	3F
marker_bit	6	

【図 5 2】

sequence_extension()	No. of bits	Mnemonic
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
profile_and_level_indication	8	uimsbf
progressive_sequence	1	uimsbf
chroma_format	2	uimsbf
horizontal_size_extension	2	uimsbf
vertical_size_extension	2	uimsbf
bit_rate_extension	12	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_extension	8	uimsbf
low_delay	1	uimsbf
frame_rate_extension_n	2	uimsbf
frame_rate_extension_d	5	uimsbf
next_start_code()		

【図 5 6】

picture_header()	No. of bits	Mnemonic
picture_start_code	32	bslbf
temporal_reference	10	uimsbf
picture_coding_type	3	uimsbf
vbv_delay	16	uimsbf
if(picture_coding_type==2 picture_coding_type==3){		
full_pel_forward_vector	1	bslbf
forward_f_code	3	bslbf
}		
if(picture_coding_type==3){		
full_pel_backward_vector	1	bslbf
backward_f_code	3	bslbf
}		
while(nextbits() == '1') {		
extra_bit_picture /*with the value '1' */	1	uimsbf
extra_information_picture	8	uimsbf
}		
extra_bit_picture /*with the value '0' */	1	uimsbf
next_start_code()		

【図 6 3】

picture_data()	No. of bits	Mnemonic
do {		
slice()		
}while(nextbits() == slice_start_code)		
next_start_code()		

【図 47】

history stream(43-5).

	bits	value
picture_display_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	7
picture_display_extension_present_flag	1	
frame_centre_horizontal_offset_1	16	
marker_bit	1	1
frame_centre_vertical_offset_1	16	
marker_bit	1	1
frame_centre_horizontal_offset_2	16	
marker_bit	1	1
frame_centre_vertical_offset_2	16	
marker_bit	1	1
frame_centre_horizontal_offset_3	16	
marker_bit	1	1
frame_centre_vertical_offset_3	16	
marker_bit	8	3F
re_coding_stream_information		
user_data_start_code	32	000001B2
re_coding_stream_info_ID	16	91EC
red_bw_flag	1	
red_bw_indicator	2	
marker_bit	5	1F
user_data		
user_data_start_code	32	000001B2
user_data	2048	
while(macroblock !=macroblock count){		
macroblock		
macroblock_address_h	8	
macroblock_address_v	8	
slice_header_present_flag	1	
skipped_macroblock_flag	1	
marker_bit	1	
macroblock_modes()	1	1
macroblock_quant	1	
macroblock_motion_forward	1	
macroblock_motion_backward	1	
macroblock_pattern	1	
macroblock_intra	1	

【図 57】

picture_coding_extension()	No. of bits	Mnemonic
extension_start_code	32	bsbf
extension_start_code_identifier	4	ulmsbf
{code[0][0] /*forward horizontal */	4	ulmsbf
{code[0][1] /*forward vertical */	4	ulmsbf
{code[1][0] /*backward horizontal */	4	ulmsbf
{code[1][1] /*backward vertical */	4	ulmsbf
intra_dc_precision	2	ulmsbf
picture_structure	2	ulmsbf
top_field_first	1	ulmsbf
frame_pred_frame_dct	1	ulmsbf
concealment_motion_vectors	1	ulmsbf
q_scale_type	1	ulmsbf
intra_vlc_format	1	ulmsbf
alternate_scan	1	ulmsbf
repeat_first_field	1	ulmsbf
chroma_420_type	1	ulmsbf
progressive_frame	1	ulmsbf
composite_display_flag	1	ulmsbf
if(composite_display_flag){		
v_axis	1	ulmsbf
field_sequence	3	ulmsbf
sub_carrier	1	ulmsbf
burst_amplitude	7	ulmsbf
sub_carrier_phase	8	ulmsbf
}		
next_start_code()		

【図 48】

history stream(43-6)

	bits	value
spatial_temporal_weight_code_flag	1	
frame_motion_type	2	
field_motion_type	2	
dct_type	1	
marker_bits	2	3
quantiser_scale_code	5	
marker_bits	3	7
PMV[0][0][0]	14	
marker_bits	2	3
PMV[0][0][1]	14	
motion_vertical_field_select[0][0]	1	
marker_bit	1	1
PMV[0][1][0]	14	
marker_bits	2	3
PMV[0][1][1]	14	
motion_vertical_field_select[0][1]	1	
marker_bit	1	1
PMV[1][0][0]	14	
marker_bits	2	3
PMV[1][0][1]	14	
motion_vertical_field_select[1][0]	1	
marker_bit	1	1
PMV[1][1][0]	14	
marker_bits	2	3
PMV[1][1][1]	14	
motion_vertical_field_select[1][1]	1	
marker_bit	1	1
coded_block_pattern	12	
marker_bits	4	F
num_mv_bits	8	
num_coef_bits	14	
marker_bits	2	3

【図 58】

extension_data()	No. of bits	Mnemonic
while(nextbits() == extension_start_code){		
extension_start_code	32	bsbf
if(!--0) /*follows sequence extension() */		
sequence_display_extension()		
/*NOTE--I never takes the value 1 because extension_data()		
never follows a group of pictures header() */		
if(!--2) /*follows picture coding extension() */		
if(nextbits() == "Quant Matrix Extension ID")		
quant_matrix_extension()		
else if(nextbits() == "Copyright Extension ID")		
copyright_extension()		
else		
picture_display_extension()		
}		

【図 62】

picture_display_extension()	No. of bits	Mnemonic
extension_start_code_identifier	4	ulmsbf
for(i=0; i; number of frame center offsets; i++){		
frame_centre_horizontal_offset	16	slmsbf
marker_bit	1	bsbf
frame_centre_vertical_offset	16	slmsbf
marker_bit	1	bsbf
}		
next_start_code()		

【図 59】

sequence_display_extension()	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
video_format	3	uimsbf
colour_description	1	uimsbf
if(colour_description)		
colour_primaries	8	uimsbf
transfer_characteristics	8	uimsbf
matrix_coefficients	8	uimsbf
)		
display_horizontal_size	14	uimsbf
marker_bit	1	bslbf
display_vertical_size	14	uimsbf
next_start_code()		
)		

【図 61】

copyright_extension()	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
copyright_flag	1	bslbf
copyright_identifier	8	uimsbf
original_or_copy	1	bslbf
reserved	7	uimsbf
marker_bit	1	bslbf
copyright_number_1	20	uimsbf
marker_bit	1	bslbf
copyright_number_2	22	uimsbf
marker_bit	1	bslbf
copyright_number_3	22	uimsbf
next_start_code()		
)		

【図 65】

macroblock()	No. of bits	Mnemonic
while(nextbits() == '0000 0001 000')		
macroblock_escape	11	bslbf
macroblock_address_increment	11-1	vlcuf
macroblock_modes()		
if(macroblock_quant)		
macroblock_quantiser_scale_code	5	uimsbf
if(red_bw_flag)		
(red_bw_flag && (red_bw_indicator ≤ 1))		
if(macroblock_motion_forward)		
(macroblock_intra && concealment_motion_vectors)		
motion_vectors(0)		
if(macroblock_motion_backward)		
motion_vectors(1)		
if(macroblock_intra && concealment_motion_vectors)		
marker_bit	1	bslbf
if(macroblock_pattern && (red_bw_flag && (red_bw_indicator == 0)))		
coded_block_pattern()		
)		

【図 72】

coded_block_pattern()	No. of bits	Mnemonic
coded_block_pattern_420	3-9	
if(chroma_format == 4:2:2)		
coded_block_pattern_1	2	uimsbf
if(chroma_format == 4:4:4)		
coded_block_pattern_2	6	uimsbf
)		

【図 60】

quant_matrix_extension()	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
load_intra_quantiser_matrix	1	uimsbf
if(load_intra_quantiser_matrix)		
intra_quantiser_matrix[64]	8*64	uimsbf
load_non_intra_quantiser_matrix	1	uimsbf
if(load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix[64]	8*64	uimsbf
load_chroma_intra_quantiser_matrix	1	uimsbf
if(load_chroma_intra_quantiser_matrix)		
chroma_intra_quantiser_matrix[64]	8*64	uimsbf
load_chroma_non_intra_quantiser_matrix	1	uimsbf
if(load_chroma_non_intra_quantiser_matrix)		
chroma_non_intra_quantiser_matrix[64]	8*64	uimsbf
next_start_code()		
)		

【図 64】

slice()	No. of bits	Mnemonic
slice_start_code	32	bslbf
slice_quantiser_scale_code	5	uimsbf
if(nextbits() == '1')		
intra_slice_flag	1	bslbf
intra_slice	1	uimsbf
reserved_bits	7	uimsbf
while(nextbits() == '1')		
extra_bit_slice /* with the value '1' */	1	uimsbf
extra_information_slice	8	uimsbf
}		
extra_bit_slice /* with the value '0' */	1	uimsbf
do {		
macroblock()		
} while(nextbits() != '000 0000 0000 0000 0000')		
next_start_code()		
)		

【図 66】

macroblock_modes()	No. of bits	Mnemonic
macroblock_type	1-9	vlcuf
if(red_bw_flag)		
(red_bw_flag && (red_bw_indicator ≤ 1))		
if(macroblock_motion_forward)		
macroblock_motion_backward		
if(picture_structure == 'frame')		
if(frame_pred_frame_dct == 0)		
frame_motion_type	2	uimsbf
else		
field_motion_type	2	uimsbf
}		
}		
if((picture_structure == "Frame picture") && (frame_pred_frame_dct == 0) && (dct_type_flag == 1) && (red_bw_flag && (red_bw_indicator ≤ 1)))		
dct_type	1	uimsbf
}		
)		

【図 67】

macroblock_type VLC code					
		macroblock_quant			
		dct_type_flag			
		macroblock_motion_forward			
		macroblock_motion_backward			
		Description			
1	0	1	0	0	Intra
01	1	1	0	0	Intra, Quant

【図 69】

macroblock_type VLC code					
		macroblock_quant			
		dct_type_flag			
		macroblock_motion_forward			
		macroblock_motion_backward			
		Description			
10	0	0	0	0	Interp, Not Coded
11	0	1	1	1	Interp, Coded
010	0	0	0	0	Bwd, Not Coded
011	0	1	0	1	Bwd, Coded
0010	0	0	0	0	Fwd, Not Coded
0011	0	1	1	0	Fwd, Coded
0001 1	0	1	0	0	Intra
0001 0	1	1	1	1	Interp, Coded, Quant
0000 11	1	1	1	0	Fwd, Coded, Quant
0000 10	1	1	0	1	Bwd, Coded, Quant
0000 01	1	1	0	0	Intra, Quant

【図 71】

motion_vector(r, s)		No. of bits	Mnemonic
motion_code[r][s][0]		1-11	vlcibf
if((!code[s][0] != 1) && (motion_code[r][s][0] != 0))			
motion_residual[r][s][0]		1-8	ulmsbf
if(dmv == 1)			
dmvector[0]		1-2	vlcibf
motion_code[r][s][1]		1-11	vlcibf
if((!code[s][1] != 1) && (motion_code[r][s][1] != 0))			
motion_residual[r][s][1]		1-8	ulmsbf
if(dmv == 1)			
dmvector[1]		1-2	vlcibf

【図 68】

macroblock_type VLC code					
		macroblock_quant			
		dct_type_flag			
		macroblock_motion_forward			
		macroblock_motion_backward			
		Description			
1	0	1	1	0	MC, Coded
01	0	1	0	0	No MC, Coded
001	0	0	0	0	MC, Not Coded
0001 1	0	1	0	0	Intra
0001 0	1	1	1	0	MC, Coded, Quant
0000 1	1	1	0	0	No MC, Coded, Quant
0000 01	1	1	0	0	Intra, Quant

【図 70】

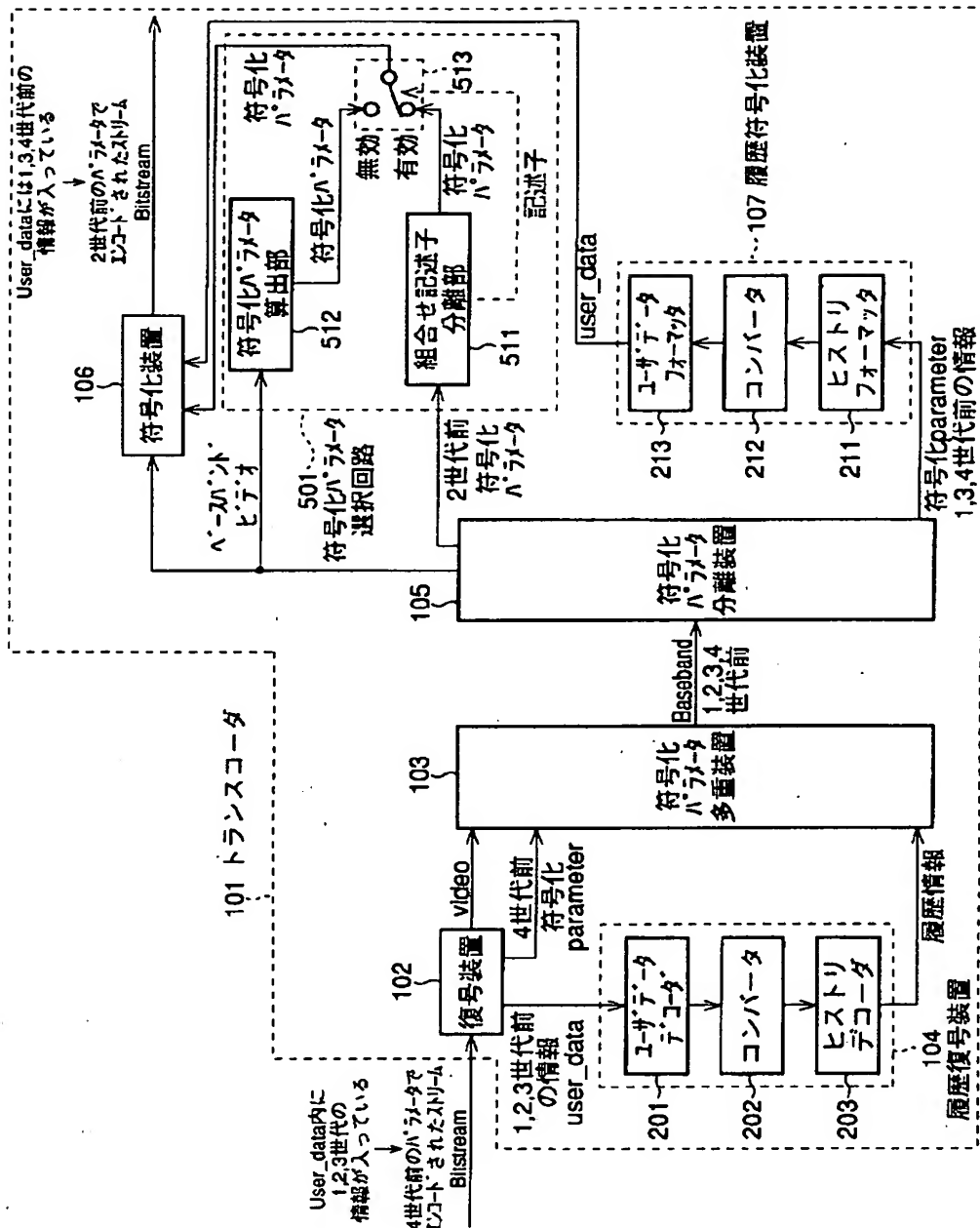
motion_vectors(s)		No. of bits	Mnemonic
if(motion_vector_count == 1)			
if((mv_format == field) && (dmv != 1))			
motion_vertical_field_select[0][s]		1	ulmsbf
motion_vector(0, s)			
else			
motion_vertical_field_select[0][s]		1	ulmsbf
motion_vector(0, s)			
motion_vertical_field_select[1][s]		1	ulmsbf
motion_vector(1, s)			

【図 74】

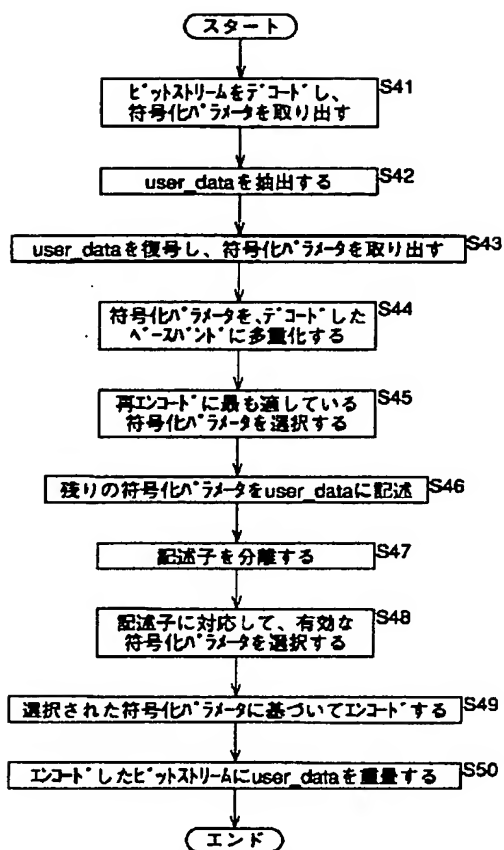
履歴情報の項目の組合せ											
num_coef_bits, num_mv_bits, num_other_bits											
q_scale_code, q_scale_type											
motion_type, mv_vert_field_sel[[]], mv[[][]]											
mb_mfwd, mb_mbwd											
mb_pattern											
coded_block_pattern											
mb_intra											
slice_start											
dct_type											
mb_quant											
skipped_mb											
組合せ1	2	2	2	2	2	2	2	2	2	2	2
組合せ2	0	2	2	2	2	2	2	2	2	2	2
組合せ3	0	2	2	2	2	0	2	1	2	1	1
組合せ4	0	2	0	1	1	0	1	1	0	1	1
組合せ5	0	0	0	0	0	0	0	0	0	0	0

履歴情報の項目の組合せ

【图 7 3】



【图 7 6】



re_coding_stream_info()	No. of bits	Microcode
user_data_start_code	32	bslbf
re_coding_stream_info_ID	16	bslbf
red_bw_flag	1	uimbsf
if(red_bw_flag)		
red_bw_indicator	2	uimbsf
if(!red_bw_flag)		
for(i=0; i<number_of_macroblock; i++)		
marker_bit	3	bslbf
num_other_bits	7	uimbsf
num_mv_bits	8	uimbsf
num_oocf_bits	14	uimbsf
}		
next_start_code()		

【图 78】

[illegible]

【図 79】

picture rate elements(79-1)

Parameter	Number format	Number of bits	Bit offset from	Bit offset to	Comments
MPEG standard flag	1bit flag	1	0	0	1==MPEG1, 0==MPEG2
red bw flag	1bit flag	1	1	1	Default= '0'
red bw indicator	2bit u	3	2	4	Default= '00'
header present flags	2bit flags	2	5	6	sequence header present flag, GOP header present flag
Extension start code flags	16 flags	16	7	22	Indicates if a given extension start code exists. The 16 flags correspond to the 16 codes defined in Table 2 of the ISO/IEC 13818-2. The flags are ordered in the order they are listed.
Other start codes	3 flags	3	23	25	user_data_start_code, sequence_end_code, sequence_and_code
sequence header					
horizontal size	14bit unsigned	14	26	39	includes extension
vertical size	14bit unsigned	14	40	53	includes extension
aspect ratio information	4bit unsigned	4	54	57	
frame rate code	4bit unsigned	4	58	61	
bit rate	30bit unsigned	30	62	91	includes extension, coded rate shall be obtained
vbv buffer size	18bit unsigned	18	92	109	includes extension
constrained parameters flag	1bit flag	1	110	110	
sequence extension					
profile and level indication	8bit unsigned	8	111	118	
progressive sequence	1bit flag	1	119	119	
chroma format	2bit unsigned	2	120	121	
low delay	1bit flag	1	122	122	
sequence display extension					
video format	3bit unsigned	3	123	125	
colour description	1bit flag	1	126	126	
colour primaries	8bit unsigned	8	127	134	
transfer characteristics	8bit unsigned	8	135	142	
matrix coefficients	8bit unsigned	8	143	150	
display horizontal size	14bit unsigned	14	151	164	
display vertical size	14bit unsigned	14	165	178	
group of pictures header					
time code	25bit field	25	179	203	
closed_gop	1bit flag	1	204	204	

【図 80】

picture rate elements(79-2)

broken link	1bit flag	1	205	205	2
picture header					
temporal reference	10bit unsigned	10	206	216	1
picture coding type	3bit unsigned	3	218	218	1
vbv delay	16bit unsigned	16	219	234	1
full pel forward vector	1bit flag	1	235	235	1
forward f code	3bit unsigned	3	236	238	1
full pel backward vector	1bit flag	1	239	239	1
backward f code	3bit unsigned	3	240	242	1
picture coding extension					
forward horizontal f code	4bit unsigned	4	243	246	1
forward vertical f code	4bit unsigned	4	247	250	1
backward horizontal f code	4bit unsigned	4	251	254	1
backward vertical f code	4bit unsigned	4	255	258	1
intra dc precision	2bit unsigned	2	259	260	1
picture structure	2bit unsigned	2	261	262	1
top field first	1bit flag	1	263	263	1
frame_pred_frame_dct	1bit flag	1	264	264	1
concealment motion vectors	1bit flag	1	265	265	1
q scale type	1bit flag	1	266	266	1
intra vlc format	1bit flag	1	267	267	1
alternate scan	1bit flag	1	268	268	1
repeat first field	1bit flag	1	269	269	1
chroma 420 type	1bit flag	1	270	270	1
progressive frame	1bit flag	1	271	271	1
composite display flag	1bit flag	1	272	272	1
v axis	1bit flag	1	273	273	1
field sequence	3bit unsigned	3	274	276	1
sub carrier	1bit flag	1	277	277	1
burst amplitude	7bit unsigned	7	278	284	1
sub carrier phase	8bit unsigned	8	285	292	1
quant matrix extension					
load intra quantiser matrix	1bit flag	1	293	293	1 (1)
load non intra quantiser matrix	1bit flag	1	294	294	1 (1)
load chroma intra quantiser matrix	1bit flag	1	295	295	1 (1)
load chroma non intra quantiser matrix	1bit flag	1	296	296	1 (1)
intra quantiser matrix(64)	64*0..255	512	297	808	2 (1)
non intra quantiser matrix(64)	64*0..255	512	809	1320	2 (1)
chroma intra quantiser matrix(64)	64*0..255	512	1321	1832	2 (1)
chroma non intra quantiser matrix(64)	64*0..255	512	1833	2344	2 (1)
picture display extension					
frame centre horizontal offset 1	16bit unsigned	16	2345	2360	2
frame centre vertical offset 1	16bit unsigned	16	2361	2376	2
frame centre horizontal offset 2	16bit unsigned	16	2377	2392	2

【図 81】

picture rate elements(79-3)

frame centre vertical offset 2	16bit unsigned	16	2393	2408	2
frame centre horizontal offset 3	16bit unsigned	16	2409	2424	2
frame centre vertical offset 3	16bit unsigned	16	2425	2440	2
copyright extension					
copyright flag	1bit flag	1	2441	2441	2
copyright identifier	8bit code	8	2442	2449	2
original or copy	1bit flag	1	2450	2450	2
copyright number	64bit unsigned	64	2451	2514	2
PTS/DTS					
PTS DTS flag	2bit flag	2	2515	2516	1
PTS Value	33bit unsigned	33	2517	2549	2
DTS Value	33bit unsigned	33	2550	2582	2
spare reserved bits					
spare	41bit unsigned	41	2583	2623	
user data area					
user data		1664	2624	4287	2
picture rate information CRC					
32-bit protection CRT	32bit unsigned	32	4288	4319	

【図 82】

D9	Cb[0][9]	Y[0][9]	Cr[0][9]	Y[1][9]	Cb[1][9]	Y[2][9]	Cr[1][9]	Y[3][9]
D8	Cb[0][8]	Y[0][8]	Cr[0][8]	Y[1][8]	Cb[1][8]	Y[2][8]	Cr[1][8]	Y[3][8]
D7	Cb[0][7]	Y[0][7]	Cr[0][7]	Y[1][7]	Cb[1][7]	Y[2][7]	Cr[1][7]	Y[3][7]
D6	Cb[0][6]	Y[0][6]	Cr[0][6]	Y[1][6]	Cb[1][6]	Y[2][6]	Cr[1][6]	Y[3][6]
D5	Cb[0][5]	Y[0][5]	Cr[0][5]	Y[1][5]	Cb[1][5]	Y[2][5]	Cr[1][5]	Y[3][5]
D4	Cb[0][4]	Y[0][4]	Cr[0][4]	Y[1][4]	Cb[1][4]	Y[2][4]	Cr[1][4]	Y[3][4]
D3	Cb[0][3]	Y[0][3]	Cr[0][3]	Y[1][3]	Cb[1][3]	Y[2][3]	Cr[1][3]	Y[3][3]
D2	Cb[0][2]	Y[0][2]	Cr[0][2]	Y[1][2]	Cb[1][2]	Y[2][2]	Cr[1][2]	Y[3][2]
D1	Cb[0][1]	Y[0][1]	Cr[0][1]	Y[1][1]	Cb[1][1]	Y[2][1]	Cr[1][1]	Y[3][1]
D0	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[0][0]	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[1][0]	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[2][0]	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[3][0]

フロントページの続き

F ターム(参考) 5C059 KK01 KK22 KK34 KK35 KK36
 MA23 MC11 MC38 ME02 NN01
 NN28 PP05 PP06 PP07 PP16
 RA01 RC11 SS07 SS11 SS20
 UA02 UA05 UA33 UA38
 5C063 AB07 AB09 AC01 BA12 CA11
 CA12

This Page is inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☒ BLACK BORDERS
- ☒ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☐ BLURED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☒ COLORED OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REPERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images
problems checked, please do not report the
problems to the IFW Image Problem Mailbox**

THIS PAGE BLANK (USPTO)